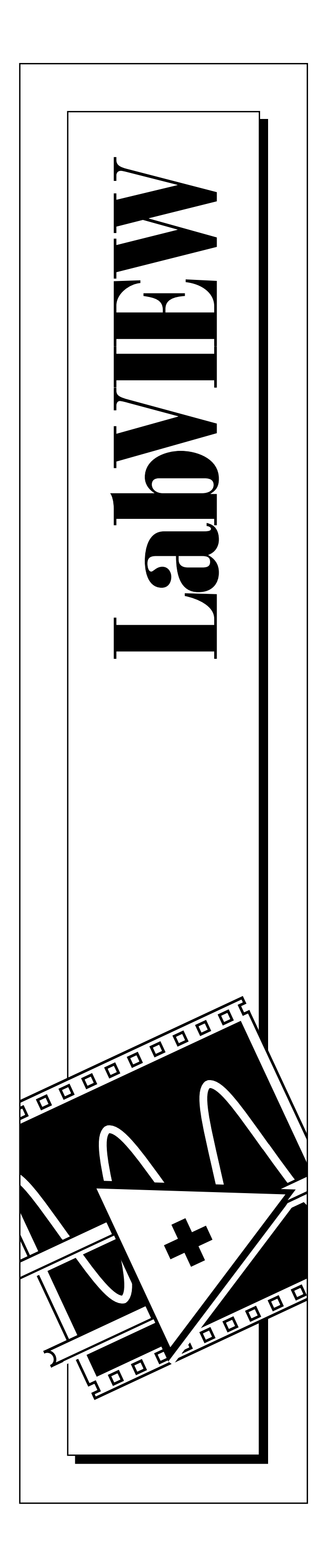


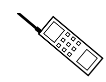
Tutorial LabVIEW®

Edition de janvier 1996
Numéro de série 321191A-01



National Instruments France

Centre d’Affaires Paris-Nord
BP 217
93153 Le Blanc-Mesnil Cedex
Tél. : (1) 48 14 24 24
Fax : (1) 48 14 24 14



Filiales

Allemagne (089) 741 31 30, Australie (03) 9 879 9422, Autriche (0662) 45 79 90 0, Belgique (02) 757 00 20, Canada (Ontario) (519) 622-9310, Canada (Québec) (514) 694-8521, Corée 596-7456, Danemark 45 76 26 00, Espagne (91) 640 0085, Finlande (90) 527 2321, France (1) 48 14 24 24, Hong Kong 2645 3186, Italie 02/48301892, Japon (03) 5472 2970, Mexique 95 800 010 0793, Norvège 32 84 84 00, Pays-Bas 0348 433466, Royaume-Uni 01635 523545, Singapour 2265886, Suède 08-730 49 70, Suisse 056 200 51 51, Taïwan 02 377 1200

National Instruments Corporate Headquarters

6504 Bridge Point Parkway Austin, TX 78730-5039 Tél.: (512) 794-0100

Informations supplémentaires

L'objectif de ce manuel est de vous aider à développer des applications en vous servant du logiciel LabVIEW. Si, malgré tout, vous avez besoin d'informations supplémentaires, vous trouverez à votre disposition les supports suivants :

Documentation LabVIEW

LabVIEW vous est fourni avec une documentation complète. Elle comprend des manuels de référence fonctionnels pour toutes les fonctions intégrées, un manuel d'utilisateur illustré d'explications et d'exemples sur les objets des faces-avant, l'environnement LabVIEW, des techniques de programmation ainsi qu'un index général pour vous permettre de localiser des renseignements particuliers.

Répertoire d'exemples

LabVIEW comprend un grand nombre d'exemples tous situés dans le répertoire `EXAMPLES`. Le VI (instrument virtuel) **readme** situé hiérarchiquement à la racine de ce répertoire vous permettra de parcourir des documents en ligne pour comprendre les exemples proposés.

Notes d'application et notes techniques

National Instruments tient gracieusement à votre disposition un très grand nombre de notes techniques et de notes d'application. Ces notes sont regroupées sur le serveur de fichiers BBS et sur le réseau Internet.

Services offerts sur support électronique

Vous avez la possibilité de vous connecter sur le serveur de fichiers BBS de LabVIEW en utilisant un modem ou le réseau Internet pour bénéficier des services suivants :

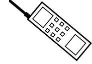
- support technique et logiciel
- correspondance électronique
- questions des utilisateurs
- publications techniques, notes d'application
- VIs utilitaires pour applications spécifiques
- mises à jour logicielles et nouveaux drivers d'instrument

Support technique en France



Télécopie

Pour bénéficier du support technique par télécopie, veuillez nous communiquer vos nom et prénom, le nom de votre société, la version de LabVIEW et le type de plate-forme sur lesquelles vous travaillez, ainsi que toutes vos questions les plus détaillées possibles.
Composez pour cela le : (1) 48 14 24 14.



Téléphone

La possibilité vous est offerte de vous entretenir directement avec nos ingénieurs d'application en composant le : (1) 48 14 24 00.



BBS (modem)

Numéro de téléphone	(1) 48 65 15 59
Débit en bauds	28800
Bits utiles	8
Bits d'arrêt	1
Parité	aucune
Adresse E-mail	`France.support@natinst.com`
Site Internet FTP	
Adresse	ftp.natinst.com
Connexion	anonyme
Mot de passe	votre adresse E-mail

Support technique international



FaxBack

Ce système de consultation d'informations automatique contient des fiches sur les produits, les questions les plus souvent posées ainsi que des notes techniques et d'application. Vous pouvez y accéder sur un combiné téléphonique à boutons-poussoirs. Les documents demandés vous sont ensuite télécopiés. Composez le : 19-1 (512) 418-1111 ou le 19-1 (800) 329-7177.



Forums Internet sponsorisés par les utilisateurs

Ces forums permettent aux utilisateurs de communiquer entre eux à propos de LabVIEW
Demandes d'abonnement : `info-labview-request@pica.army.mil`
Forums : `info-labview@pica.army.mil`



Adresses E-mail

Il s'agit de boîtes à lettres de support technique gérées par les ingénieurs d'application :
`lv.support@natinst.com`
`lw.support@natinst.com`
`hiq.support@natinst.com`
`gpib.support@natinst.com`
`daq.support@natinst.com`
`vxi.support@natinst.com`

Informations importantes

Limitations de garantie

Les disquettes sur lesquelles vous recevez les logiciels de National Instruments sont garanties contre les pannes survenant lors de l'exécution de programmes, qui seraient dues à des défauts matériels ou de fabrication. La période de cette garantie est de 90 jours à partir de la date de livraison, attestée par les reçus ou autres documents. Le cas échéant, National Instruments corrigera ou remplacera la disquette qui ne permettrait pas l'exécution normale des programmes, à condition qu'un tel défaut soit stipulé au cours de la période de garantie. National Instruments ne garantit pas que le fonctionnement de ses logiciels ne sera pas interrompu ni se déroulera sans erreur.

Un numéro RMA (Return Material Authorization) d'autorisation de retour de matériel doit être délivré par l'usine et clairement indiqué sur l'emballage du produit afin que celui-ci puisse être accepté pour la réparation sous garantie. National Instruments prendra à sa charge les frais de transport pour renvoyer à son propriétaire les éléments couverts par la garantie.

National Instruments estime avoir fait tout ce qu'il fallait pour que les informations contenues dans ce manuel soient exactes. Ce document a été soigneusement relu pour la précision des informations techniques qu'il contient. Au cas où il resterait malgré tout des erreurs techniques ou des fautes typographiques, National Instruments se réserve le droit d'apporter des modifications aux futures éditions de ce document sans préavis aux détenteurs de cette édition. Le lecteur est prié de prendre contact directement avec National Instruments s'il suspecte des erreurs. En aucun cas, National Instruments ne pourra être tenu responsable des problèmes liés à l'utilisation de ce document ou aux informations qu'il contient.

A L'EXCEPTION DE CE QUI EST SPECIFIE ICI, NATIONAL INSTRUMENTS N'ACCORDE AUCUNE AUTRE GARANTIE, EXPLICITE OU IMPLICITE, ET REJETTE PARTICULIEREMENT TOUTE GARANTIE LIEE A L'ACTE DE VENTE ET A L'ADEQUATION DE SES PRODUITS A UN BESOIN PARTICULIER. LES DROITS DES UTILISATEURS A RECOUVRER LES DOMMAGES CAUSES PAR UNE FAUTE OU NEGLIGENCE DE LA PART DE NATIONAL INSTRUMENTS SERONT LIMITES AUX SOMMES VERSEES PAR L'UTILISATEUR. NATIONAL INSTRUMENTS NE SERA PAS PASSIBLE DE DOMMAGES ET INTERETS A LA SUITE DE PERTES DE DONNEES OU DE PROFITS, OU DE TOUS DOMMAGES (ACCIDENTELS OU NON) LIES A L'UTILISATION DE SES PRODUITS, MEME S'IL EN AVAIT ETE PREALABLEMENT AVERTI. Cette limitation de la responsabilité de National Instruments s'appliquera quelles que soient la nature et l'origine du préjudice, que ce soit à la suite d'un contrat ou la conséquence d'un acte délictueux, y compris par négligence. Toute action contre National Instruments doit être conduite dans l'année qui suit la cause de cette action. National Instruments ne pourra être tenu responsable de tout retard en performance dû à des causes qui iraient au-delà de ce qu'il lui est raisonnablement possible de faire. La garantie fournie ici ne couvre pas les dommages, défauts, dysfonctionnements, ou défauts de service dus à des erreurs commises par l'utilisateur dans l'interprétation des instructions de National Instruments en ce qui concerne l'installation, le fonctionnement et la maintenance. Elle ne couvre pas non plus les négligences, les modifications ou mauvais usages du produit de la part de l'utilisateur, les chutes de tension ou surintensités, le feu, les inondations, les accidents, les agissements de tierces personnes, et tout autre événement incontrôlable.

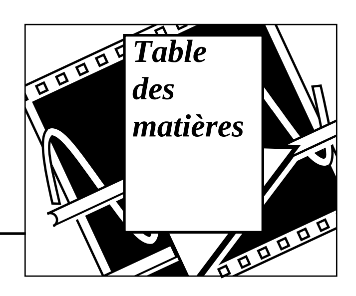
Copyright

Marques déposées

LabVIEW ® et NI-488M™ sont des marques déposées de National Instruments Corporation. Les produits et noms de sociétés cités sont des marques déposées par leurs propriétaires respectifs.

MISE EN GARDE CONCERANT L'UTILISATION DES PRODUITS DE NATIONAL INSTRUMENTS DANS LES APPLICATIONS MÉDICALES ET HOSPITALIERES

Les produits de National Instruments ne sont pas conçus avec des composants et suivant des méthodes de tests prévus pour assurer un niveau de fiabilité convenant à leur utilisation dans les applications de traitement et de diagnostic sur les personnes. Les applications des produits de National Instruments impliquant des traitements médicaux ou cliniques peuvent potentiellement occasionner des blessures accidentelles à cause d'une panne des produits, ou à cause d'une erreur de la part de l'utilisateur ou du concepteur de l'application. Toute utilisation ou application des produits National Instruments pour ou dans des traitements médicaux ou cliniques doit être effectuée par un personnel médical correctement formé et qualifié, et toutes les garanties médicales d'usage, tous les équipements et toutes les procédures qui sont appropriés à cette situation particulière pour éviter les blessures graves ou la mort, doivent toujours être mis en œuvre lorsque l'on utilise des produits de National Instruments. Les produits de National Instruments N'ONT PAS été conçus pour se substituer à toute forme de procédé, procédure ou équipement utilisée pour la surveillance médicale ou pour garantir la santé publique dans les traitements médicaux et cliniques.



Avant-propos

Chapitre 1
Introduction à LabVIEW

Chapitre 2
La création d'un sous-VI

Chapitre 3
Boucles et graphes déroulants

Chapitre 4
Tableaux, clusters et graphes

Chapitre 5
Structures Condition, structures Séquence et boîte de calcul

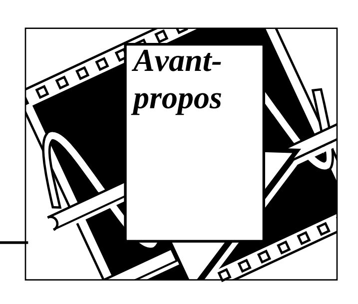
Chapitre 6
Chaînes de caractères et E/S sur fichier

Chapitre 7
La personnalisation des VIs

Chapitre 8
L'acquisition de données et le contrôle d'instruments

Chapitre 9
Les techniques et astuces de programmation et de mise au point

Chapitre 10
La conception des programmes



Le *Tutorial LabVIEW* contient les informations dont vous avez besoin pour exploiter les principales fonctions du logiciel LabVIEW (*Laboratory Virtual Instrument Engineering Workbench*). LabVIEW simplifie le développement des applications de traitements scientifiques, de contrôle/commande de processus industriels, de test et mesure électroniques, et de bien d'autres applications en programmation.

Ce manuel vous offre un aperçu des concepts de base de LabVIEW, illustré d'exercices vous permettant d'apprendre ce que vous devez savoir pour développer vos propres instruments virtuels (VIs) le plus rapidement possible. Ce manuel ne reprend pas pour autant toutes les caractéristiques de LabVIEW. Il s'articule d'abord autour des caractéristiques les plus importantes de LabVIEW dans le souci de vous aider dans vos tâches de programmation.

Dans ce manuel, nous partons du principe que vous savez parfaitement vous servir de votre ordinateur personnel et que vous en maîtrisez son système d'exploitation.

Pour une meilleure cohérence entre ce tutorial et le logiciel, certains termes anglais ont été délibérément conservés, notamment ceux qui apparaissent dans les menus et palettes.

Organisation du manuel

Chaque chapitre traite d'un concept particulier de LabVIEW. Mais sachez que les VIs que vous développerez mettront souvent en œuvre plusieurs concepts à la fois. C'est pourquoi nous vous conseillons de parcourir l'ensemble du tutorial avant de commencer à construire votre application.

Certains chapitres de ce tutorial comportent une section intitulée *Quelques informations supplémentaires*, qui vous offre un aperçu des caractéristiques avancées de LabVIEW et vous renvoie à d'autres documents et exemples de VIs.

Le manuel s'organise de la façon suivante :

- Le chapitre 1, *Introduction à LabVIEW*, décrit ce qu'est LabVIEW, ce qu'est un Instrument Virtuel (VI), comment utiliser l'environnement de LabVIEW (fenêtres, menus, palettes et outils), et vous apprend à faire fonctionner les VIs, les éditer et les créer.
- Le chapitre 2, *La création d'un sous-VI*, traite des sous-VIs, vous apprend à créer leurs icônes et leurs connecteurs et à utiliser les VIs comme des sous-VIs.
- Le chapitre 3, *Boucles et graphes déroulants*, vous présente les boucles *While*, vous apprend à afficher des données sous forme de graphes déroulants, vous renseigne sur les registres à décalage et leur utilisation et vous apprend à utiliser les boucles *For*.
- Le chapitre 4, *Tableaux, clusters et graphes*, traite de la création de tableaux et de l'utilisation des fonctions élémentaires dans les tableaux, les *clusters* et les graphes. Vous y découvrirez également le principe du polymorphisme et la façon d'utiliser les graphes pour afficher des données.
- Le chapitre 5, *Structures Condition, structures Séquence et boîte de calcul*, explique comment utiliser les structures de programmation dites Condition et Séquence et la possibilité d’exploiter les boîtes de calcul (*formula nodes*) classiques.
- Le chapitre 6, *Chaînes de caractères et E/S sur fichier*, vous apprend à créer des commandes et des indicateurs de chaînes de caractères, présente les fonctions de chaînage ainsi que les opérations d'Entrées/Sorties sur fichier. Vous apprendrez aussi à enregistrer les données dans des feuilles de calcul et à entrer et à lire des données dans les fichiers texte.
- Le chapitre 7, *La personnalisation des VIs*, vous explique comment utiliser les options de configuration des VIs et sous-VIs et comment personnaliser les commandes et les indicateurs.
- Le chapitre 8, *L’acquisition de données* (pour Windows, Macintosh et Sun) *et le contrôle d'instruments*, vous explique comment acquérir des données à partir d'une carte d'acquisition de données, traite de la mise en œuvre de VISA et du GPIB, vous apprend à contrôler une interface de port série à partir de LabVIEW, traite du contrôle d'instrumentation VXI (pour Windows, Macintosh et Sun), explique ce que sont les drivers d'instrument, comment les utiliser et enfin vous explique comment mettre en œuvre un VI **Frequency Response Test**.

- Le chapitre 9, *Les techniques et astuces de programmation et de mise au point*, vous apprend les techniques d'édition de programmes et vous fournit quelques astuces pour faciliter le développement et la mise au point de votre application.
- Le chapitre 10, *La conception des programmes*, vous présente des techniques utiles à la conception de programmes ainsi que des suggestions de styles de programmation.
- Le chapitre 11, *Développements supplémentaires*, contient des informations sur d'autres sources utiles à examiner au fur et à mesure que vous construisez des applications LabVIEW.
- L'annexe, *La parole est à vous*, contient des formulaires que vous pouvez utiliser pour solliciter l'assistance de National Instruments ou pour nous faire part de vos remarques sur les produits et les manuels de la société.
- Le *Glossaire* contient une liste alphabétique des termes utilisés dans ce manuel, notamment les abréviations, acronymes, préfixes des unités de mesures, caractères mnémoniques et symboles.
- L'*Index* contient la liste alphabétique des termes et des sujets principaux abordés dans ce tutorial, avec le numéro de page auquel se référer.

Conventions d'écriture et abréviations

La rédaction de ce manuel s'appuie sur les conventions suivantes :

gras — Le texte en gras sert à identifier des éléments des menus, des boutons ou des choix de boîtes de dialogue. Il sert aussi à identifier les paramètres d'entrée et de sortie des VIs.

italique — L'italique sert à mettre en valeur un mot, à présenter une référence croisée, ou un concept essentiel. Certains mots conservés en anglais sont également en italique.

gras italique — Le gras italique souligne une remarque, un avertissement ou une mise en garde.

`caractères courier` — Une police Courier désigne du texte à taper au clavier. Elle sert aussi à présenter des scripts de programmes, des messages et des réponses que l'ordinateur affiche à l'écran.

`caractères courier italique` — L'italique utilisé dans cette police indique que vous devez fournir les termes ou valeurs appropriés à la place de ces articles.

< > — Les parenthèses angulaires encadrent le nom d'une touche du clavier. Exemple : <`Maj`>.

- — Un trait d'union séparant deux ou plusieurs noms de touches entre parenthèses angulaires signifie qu'il faut appuyer simultanément sur les touches mentionnées. Exemple : <`Maj-Suppr`> .

» — Le symbole » vous aide à vous déplacer au sein des menus et des boîtes de dialogue jusqu'à l'exécution de la dernière action. La séquence

File»Page Setup»Options»Substitute Fonts

vous invite à ouvrir le menu **File**, à choisir l'option **Page Setup**, à sélectionner **Options** puis, pour finir, à sélectionner l'option **Substitute Fonts** dans la dernière boîte de dialogue.

chemins — Dans ce manuel, les chemins sont indiqués avec des barres obliques inverses (\) pour séparer les noms des lecteurs, répertoires et fichiers, comme dans l'exemple suivant : `lecteur\rép1\rép2\monfich`

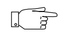
IEEE 488.1 et IEEE 488.2 — IEEE 488.1 et IEEE 488.2 se rapportent respectivement aux normes ANSI/IEEE 488.1-1987 et ANSI/IEEE 488.2-1987, qui définissent le GPIB.



Mise en garde : ***Cette icône située à gauche d'un texte en gras italique annonce une mise en garde. Elle vous alerte sur des risques éventuels pour vous ou votre matériel.***



Avertissement : ***Cette icône située à gauche d'un texte en gras italique annonce un avertissement. Elle vous alerte sur un risque de perte de données ou de défaillance du système.***



Remarque : ***Cette icône située à gauche d'un texte en gras italique introduit une remarque importante.***

Les abréviations, acronymes, préfixes pour les unités de mesures, caractères mnémoniques, symboles, ainsi que d'autres termes, sont répertoriés dans le *Glossaire*.

Références bibliographiques

Les documents mentionnés ci-après contiennent des informations susceptibles de vous aider lors de la lecture de ce manuel.

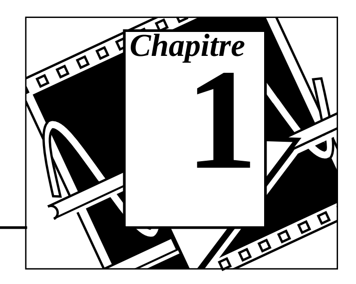
- *LabVIEW Analysis VI Reference Manual*
- *LabVIEW Code Interface Reference Manual*
- *LabVIEW Communication VI Reference Manual*
- *LabVIEW Data Acquisition Basics Manual* ***(Windows, Macintosh et Sun)***
- *LabVIEW Data Acquisition VI Reference Manual* ***(Windows, Macintosh et Sun)***
- *LabVIEW Instrument I/O VI Reference Manual*
- *LabVIEW User Manual*
- Norme ANSI/IEEE 488.1-1987, *IEEE Standard Digital Interface for Programmable Instrumentation*
- Norme ANSI/IEEE 488.2-1987, *IEEE Standard Codes, Formats, Protocols, and Common Commands*
- *LabVIEW Function Reference Manual* est disponible sur disquette ; la version imprimée est disponible sur demande.

La parole est à vous

Nous souhaitons recevoir vos commentaires sur nos produits et les manuels correspondants. Nous sommes également intéressés par les applications que vous développez avec nos produits, et sommes prêts à vous aider si vous rencontrez des problèmes lors de leur utilisation. Pour vous faciliter la tâche, si vous souhaitez nous contacter, ce manuel contient des formulaires, situés dans l'annexe, *La parole est à vous*, à la fin du manuel.

Cours de formation

National Instruments vous propose des cours élémentaires de formation ainsi que des cours de perfectionnement pour vous aider à maîtriser rapidement LabVIEW et à développer vos applications avec succès. Le Cours Elémentaire vous enseigne non seulement les principes fondamentaux de LabVIEW, mais il vous apprend également à développer des applications d'acquisition de données et de contrôle d'instruments. Quant au Cours de perfectionnement, il vous apprend à augmenter les performances et l'efficacité de vos applications développées avec LabVIEW. N'hésitez pas à contacter National Instruments pour obtenir le catalogue complet des cours, avec les tarifs et les dates correspondants.





Introduction à LabVIEW

Ce chapitre décrit ce qu’est LabVIEW, ce qu’est un Instrument Virtuel (VI), comment utiliser l’environnement LabVIEW (les fenêtres, les menus, les palettes et les outils), comment exécuter les VIs, les éditer et les créer.

LabVIEW étant un outil de développement de programmes aux possibilités particulièrement étendues, ce tutorial ne prétend pas vous expliquer comment résoudre tous les problèmes liés à la programmation. Il vous enseigne plutôt les principes fondamentaux de LabVIEW, vous familiarise avec les outils de programmation du logiciel et vous fournit des exemples pratiques d’utilisation, ceux-ci se voulant représentatifs des problèmes de programmation auxquels vous serez probablement confronté.

Si, après avoir étudié ce manuel, vous souhaitez une formation complémentaire, sachez que National Instruments propose des cours de formation pour vous aider à maîtriser rapidement LabVIEW et à développer des applications opérationnelles.

L’assimilation du Cours Elémentaire de LabVIEW vous familiarisera avec les aspects fondamentaux du logiciel, tout en vous offrant la possibilité de développer vous-même des applications concrètes de contrôle d’instruments et d’acquisition de données (pour les plates-formes Windows, Macintosh et Sun). Si vous décidez ensuite de suivre le cours de perfectionnement de LabVIEW, vous apprendrez à optimiser les performances et l’efficacité des applications du logiciel, tout en assimilant ses caractéristiques les plus avancées.

Pour obtenir le catalogue complet des cours de formation, la date des sessions ainsi que les tarifs, veuillez vous reporter à l’adresse figurant sur la deuxième page de couverture de ce manuel pour contacter National Instruments.

Informations relatives au chapitre

Au début de chaque chapitre, vous trouverez un encadré tel que celui-ci, dans lequel sont indiqués les différents points abordés dans le chapitre considéré.

Vous allez apprendre :

- Ce qu'est LabVIEW.
- Ce qu'est un Instrument Virtuel (VI).
- Comment utiliser l'environnement LabVIEW (fenêtres et palettes).
- Comment exécuter des VIs.
- Comment éditer des VIs.
- Comment créer des VIs.

Qu'est-ce que LabVIEW ?

LabVIEW est un logiciel de développement d'applications, comparable à la plupart des systèmes de développement en langage C ou BASIC disponibles sur le marché, ou encore à LabWindows de National Instruments. Cependant, LabVIEW se distingue des autres logiciels sur au moins un point important. En effet, la majorité d'entre eux s'articulent autour de langages à base de texte dont la programmation consiste à empiler des lignes de code, tandis que LabVIEW utilise un langage de programmation graphique, le *langage G*, pour créer un programme sous forme de diagramme.

Inutile d'être expert en programmation pour pouvoir utiliser LabVIEW. La terminologie, les icônes et les principes inhérents à LabVIEW, tous familiers aux ingénieurs et aux scientifiques, font appel à des symboles graphiques pour décrire les opérations de programmation.

LabVIEW offre des bibliothèques étendues de fonctions et de routines (blocs pré-programmés) capables de répondre à la plupart des besoins en programmation. En ce qui concerne les plates-formes Windows, Macintosh et Sun, LabVIEW comprend également des bibliothèques de fonctions spécifiques à l'acquisition de données et au pilotage d'instruments VXI et GPIB, ou encore d'instruments connectés sur une simple liaison série. Il existe aussi des bibliothèques dédiées à la

présentation, à l’analyse et au stockage des données. LabVIEW intègre une panoplie complète d’outils de développement de programme conventionnels, de sorte que vous pouvez définir des points d’arrêt, animer l’exécution du programme en mettant en évidence le cheminement des données et exécuter pas à pas votre programme. Le développement et la mise au point du programme s’en trouvent ainsi facilités.

Comment fonctionne LabVIEW ?

LabVIEW comprend des bibliothèques de fonctions et des outils de développement spécialement conçus pour les applications de contrôle d’instruments. Pour les plates-formes Windows, Macintosh et Sun, LabVIEW intègre également des bibliothèques de fonctions et des outils de développement pour les applications d’acquisition de données. Un programme LabVIEW est appelé *instrument virtuel* (VI) tout simplement parce que sa représentation et son fonctionnement ressemblent à ceux des instruments classiques. Néanmoins, les VIs diffèrent en ce sens qu’ils tirent leur fonctionnalité de la programmation informatique. Ils offrent une interface utilisateur interactive avec l’équivalent en code source, et acceptent les paramètres des VIs de niveau supérieur. Ces points méritent une explication complémentaire.

- Un VI intègre une interface utilisateur interactive appelée *face-avant*, puisqu’elle simule la face-avant d’instruments physiques. La face-avant contient des boutons rotatifs, des boutons-poussoirs, des graphes et autres commandes et indicateurs. Vous saisissez les données à l’aide du clavier ou de la souris, puis vous visualisez les résultats à l’écran.
- Un VI reçoit des instructions de son *diagramme*, que vous construisez en langage graphique G. Le diagramme, qui correspond au code source du VI, réduit ainsi la programmation à une simple manipulation graphique.
- Le VI présente une structure hiérarchique et modulaire. Vous pouvez l’utiliser comme un programme principal ou comme un sous-programme à l’intérieur d’autres programmes ou de sous-programmes. Un VI contenu à l’intérieur d’un autre VI s’appelle un *sous-VI*. Le *cadre icône/connecteur* d’un VI répertorie sous forme graphique tous ses paramètres si bien que d’autres VIs peuvent lui transmettre des données, en le considérant comme un sous-VI.

Avec de telles caractéristiques, LabVIEW adhère entièrement au concept de *programmation modulaire*. Ainsi, vous scindez une application en une série de tâches que vous pouvez subdiviser autant de fois que nécessaire jusqu'à ce qu'une application complexe soit ramenée à une série de tâches élémentaires, faciles à mettre en œuvre. Vous construisez un VI pour chaque tâche puis rassemblez les VIs ainsi réalisés dans un diagramme pour leur faire exécuter une tâche plus complexe. Au final, votre VI principal contient un ensemble de sous-VIs qui représentent les fonctions de l'application.

En tant qu'entité opérationnelle à part entière, chaque VI peut fonctionner indépendamment du reste de l'application, ce qui facilite la mise au point de celle-ci. En outre, de nombreux sous-VIs de bas niveau sont souvent réutilisables dans plusieurs applications. Vous pouvez donc développer une gamme de sous-VIs spécialisés, adaptés aux applications que vous envisagez de réaliser.

Installation de LabVIEW

Pour en savoir plus sur la procédure d'installation de LabVIEW, veuillez vous reporter à la notice fournie avec votre version LabVIEW.

Si, lors de l'installation, vous avez choisi la configuration par défaut, vous pouvez travailler sans problème avec le manuel que vous avez entre les mains. Si vous souhaitez ensuite explorer les diverses options de configuration proposées par LabVIEW, veuillez vous reporter à la section *Préférences Boîtes de dialogue* du chapitre 8, *La personnalisation de l'environnement LabVIEW*, du *Manuel de l'utilisateur LabVIEW*.

Les fichiers de LabVIEW

Le système LabVIEW se compose de l'application proprement dite et d'un certain nombre de fichiers associés.

LabVIEW utilise plusieurs répertoires et fichiers du disque dur pour stocker les informations nécessaires à la création de vos VIs. Parmi ces répertoires et fichiers, citons notamment :

- Le répertoire `vi.lib` qui contient les bibliothèques de VIs, comme par exemple les VIs d'analyse.
- Le répertoire `examples` qui rassemble de nombreux exemples de VIs illustrant les fonctionnalités de LabVIEW.

- La bibliothèque `tutorial.llb`, située dans le répertoire `vi.llb`, qui comprend les VIs décrits dans ce tutorial.

Vous accédez au contenu de ces fichiers et de ces répertoires à partir de LabVIEW.

Les Instruments Virtuels

Les programmes LabVIEW sont appelés des Instruments Virtuels (VIs). Ces VIs se composent de trois éléments de base : la *face-avant*, le *diagramme*, et l'*icône/connecteur*.

OBJECTIF Ouvrir, examiner et manipuler un VI afin de se familiariser avec les concepts de base d'un instrument virtuel.

La face-avant

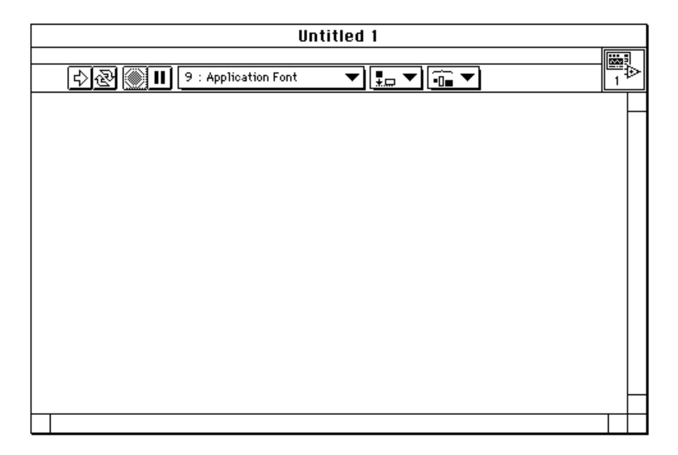
1. ***(Windows)*** Ouvrez LabVIEW en double-cliquant avec le bouton de la souris sur l'icône d'application LabVIEW. Si c'est la première fois que vous lancez LabVIEW, le programme vous invite à saisir votre nom, celui de votre société ainsi que le numéro de série de votre logiciel.

   (Macintosh) Lancez LabVIEW en double-cliquant sur l'icône LabVIEW du dossier LabVIEW. Si c'est la première fois que vous lancez LabVIEW, le programme vous invite à saisir votre nom, celui de votre société ainsi que le numéro de série de votre logiciel.

   (UNIX) Lancez LabVIEW en tapant `labview <Return>` dans une fenêtre vierge. Si LabVIEW ne figure pas dans votre chemin exécutable, il vous faudra taper le chemin de l'exécutable LabVIEW suivi de `labview`, comme dans l'exemple suivant :

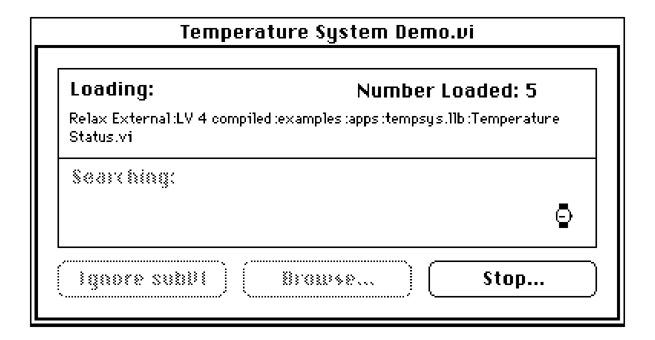
   ```
   /usr/lib/labview/labview
   ```

(Toutes les plates-formes) Après quelques instants, une face-avant vierge sans titre apparaît.

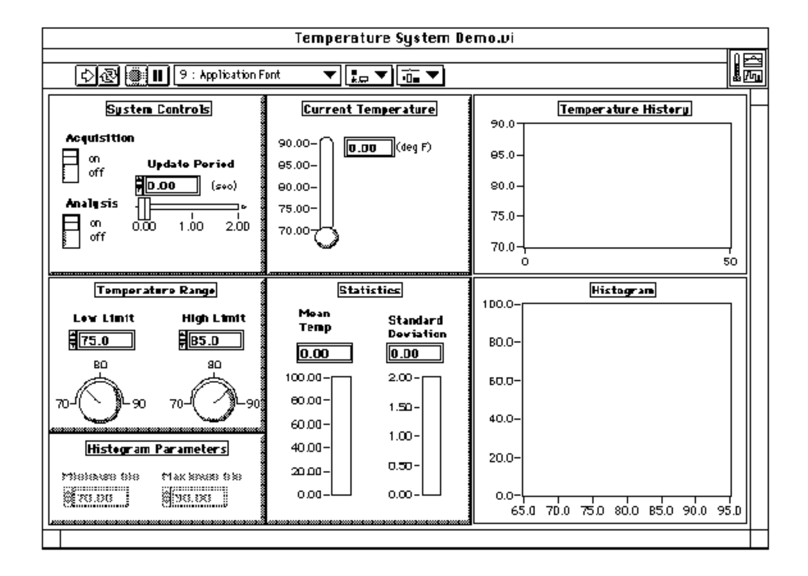


2. Ouvrez le VI **Temperature System Demo** en procédant de la manière suivante :
   a. Sélectionnez **File»Open**.
   b. Double-cliquez sur `examples`, puis sur `apps` et `tempsys.llb.`
   c. Double-cliquez sur `Temperature System Demo.vi`.

Pendant le chargement du VI, une boîte de dialogue apparaît à l'écran, indiquant le nom du VI en cours de chargement, le nom du disque dur sur lequel se trouve le VI, les répertoires et les chemins recherchés ainsi que le numéro du VI en cours de chargement. L'illustration ci-après présente la boîte de dialogue qui apparaît lorsque vous chargez le VI **Temperature System Demo**.



Après quelques instants, la fenêtre **Temperature System Demo** apparaît comme dans l’illustration suivante. La face-avant comporte plusieurs commandes numériques, des interrupteurs tout-ou-rien, des commandes curseurs, des boutons de commande rotatifs, des graphes, des graphes déroulants et un indicateur en forme de thermomètre.

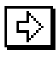


La barre d'outils de la face-avant

La face-avant possède une barre d'outils composée de boutons et d'indicateurs d'état permettant d'exécuter et de mettre au point les VIs. Vous pouvez également y choisir les polices de caractères, l'alignement et la répartition pour l'édition des VIs.

Barre d'outils de la face-avant :

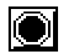




bouton Exécution : exécute le VI



bouton Exécution permanente : exécute le VI en permanence, ce qui est très utile pour la mise au point



bouton Stop : interrompt l'exécution du VI



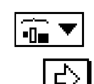
bouton Pause/Reprise : interrompt provisoirement l'exécution du VI/ Reprend l'exécution du VI



menu **Police de caractères** : définit les options relatives aux polices, y compris le type, la taille, le style et la couleur (des polices)



menu **Alignement** : définit les options relatives à l'alignement, y compris l'alignement vertical, la marge supérieure, gauche, etc., pour deux ou plusieurs objets



menu **Distribution** : définit les options relatives à la répartition, y compris les écarts, la compression, etc., pour deux ou plusieurs objets

1. Dans la face-avant, exécutez le VI en cliquant sur le bouton Exécution de la barre d'outils.

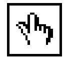


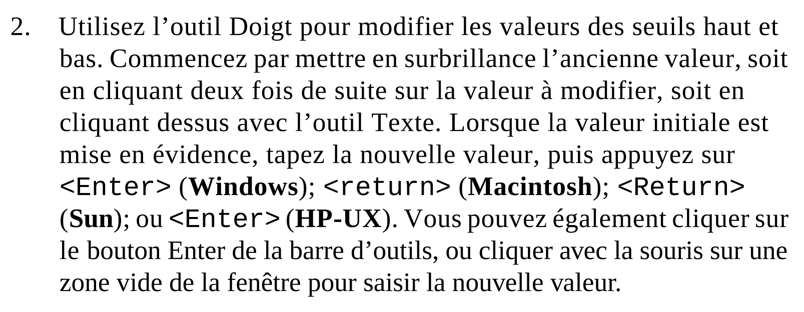
Le bouton change alors d'aspect pour indiquer que le VI est en cours d'exécution.

Le VI **Temperature System Demo** simule une application de contrôle de température. Il relève les mesures de température puis les affiche sur le thermomètre ainsi que dans le graphe déroulant. Le curseur **Update Period** contrôle la vitesse à laquelle le VI lit la valeur de la température. LabVIEW trace également les seuils haut et bas de la température sur le graphe déroulant que vous pouvez modifier en utilisant les boutons rotatifs Temperature Range, situés au centre de la

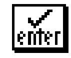
bordure gauche. Si la mesure de température relevée est en dehors des seuils prédéfinis, des voyants lumineux s’allument à côté du thermomètre.

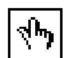
Le VI continue à s’exécuter jusqu’à ce que vous cliquiez sur le bouton Acquisition pour le désactiver. Vous pouvez aussi activer et désactiver l’analyse des données. L’analyse consiste à calculer en permanence la valeur moyenne et l’écart-type de la température ainsi qu’à établir un histogramme de ces valeurs de température.



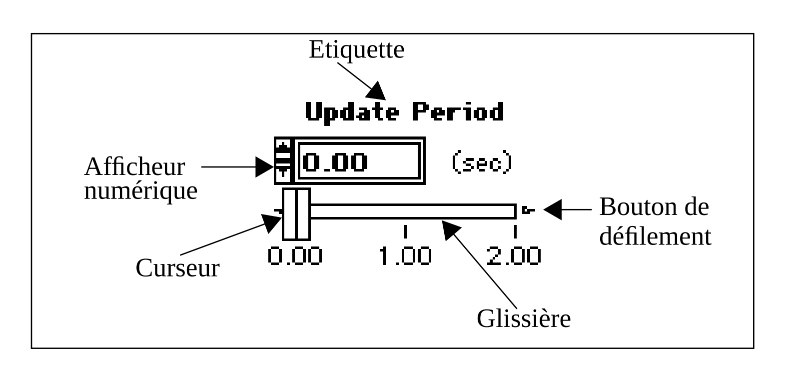


2. Utilisez l’outil Doigt pour modifier les valeurs des seuils haut et bas. Commencez par mettre en surbrillance l’ancienne valeur, soit en cliquant deux fois de suite sur la valeur à modifier, soit en cliquant dessus avec l’outil Texte. Lorsque la valeur initiale est mise en évidence, tapez la nouvelle valeur, puis appuyez sur `<Enter>` (**Windows**); `<return>` (**Macintosh**); `<Return>` (**Sun**); ou `<Enter>` (**HP-UX**). Vous pouvez également cliquer sur le bouton Enter de la barre d’outils, ou cliquer avec la souris sur une zone vide de la fenêtre pour saisir la nouvelle valeur.



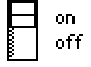


3. Modifiez le curseur **Update Period**, représenté dans l’illustration suivante, en plaçant l’outil Doigt sur le curseur, en cliquant dessus et en le faisant glisser ailleurs.





4. Exercez-vous à régler les autres commandes.



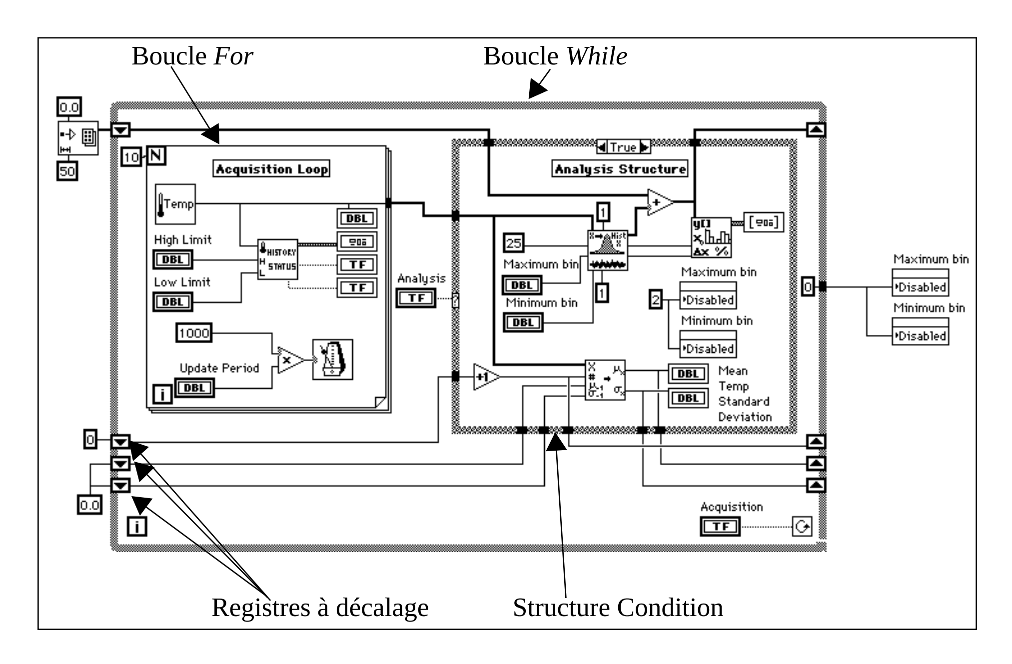
5. Arrêtez le VI en cliquant sur l’interrupteur Acquisition. Il est possible que le VI ne s’arrête pas immédiatement car il doit d’abord attendre que la dernière équation ou que le dernier jeu d’analyse soit terminé(e).

Remarque : ***il est préférable de laisser l'exécution du VI aller à son terme, ou alors de prévoir une méthode pour l'arrêter, par exemple en plaçant comme ici un interrupteur sur la face-avant. Dans le cas présent, le VI prend en compte les dernières valeurs de température, les analyse et termine seulement son exécution lorsque vous cliquez sur l'interrupteur Acquisition.***

Même si l'exécution du VI s'interrompt en cliquant sur le bouton Stop de la barre d'outils, ce n'est pas pour autant la meilleure façon d'arrêter les VIs dans la mesure où le bouton Stop interrompt immédiatement le programme. Vous risquez ainsi d'interrompre des fonctions E/S cruciales, ce qui n'est jamais souhaitable.

Le diagramme

Le diagramme suivant représente une application LabVIEW complète. Il est un parfait exemple du niveau de complexité potentiel de la programmation LabVIEW. Les prochains chapitres de ce tutorial reprennent en détail les structures et les éléments abordés dans cette section. Pour l'instant, il n'est pas nécessaire de connaître la signification des différents éléments du diagramme pour en apprécier le principe.



1. Ouvrez le VI **Temperature System Demo** en choisissant **Windows»Show Diagram.**
2. Etudiez les différents objets présents dans le diagramme.

A chaque face-avant correspond un diagramme, qui équivaut en fait au programme du VI. Vous construisez ce diagramme en utilisant le langage de programmation graphique G. Vous devez alors le considérer comme le code source. Certains éléments du diagramme représentent ce qu'il est convenu d'appeler les nœuds du programme comme les boucles *For*, les structures Condition et les fonctions de multiplication. Tous ces éléments sont *câblés* entre eux de façon à suivre le flux des données dans le diagramme.

La structure la plus externe est la boucle *While*. Elle contient l'ensemble des éléments du diagramme et se charge d'exécuter tout ce qui s'y trouve jusqu'à ce que l'interrupteur Acquisition soit placé en position ARRET. Les flèches figurant sur cette boucle *While* s'appellent des *registres à décalage* et servent à stocker les données d'un cycle de boucle à l'autre. Les valeurs stockées dans les registres à décalage sont, de haut en bas, l'histogramme, la valeur d'itération de l'analyse, la valeur moyenne et l'écart-type.

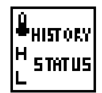
Les deux structures principales à l'intérieur de la boucle *While* sont la boucle *For* et la structure Condition, l'acquisition de données se faisant par l'intermédiaire de la boucle *For*. La boucle *For* réalise 10 relevés de température au rythme indiqué par la période de rafraîchissement **Update Period** et restitue le tracé de chacun des relevés sur le thermomètre et le graphe déroulant. Le VI compare également la température obtenue par rapport aux seuils haut et bas.

La structure Condition gère l'analyse de la température. Si le commutateur **Analysis** est sur la position ARRET, le VI n'effectue aucune analyse. Pour vous en rendre compte, vous pouvez cliquer sur l'une des flèches situées en regard du mot `True`. Si c'est FALSE, aucune analyse n'est effectuée et les valeurs d'itération d'analyse et d'histogramme sont remises à zéro. Revenez sur TRUE en procédant de la même façon qu'auparavant. Cette fois-ci, les données sont analysées par deux sous-VIs : l'un fournit la valeur moyenne dynamique et l'écart-type des températures tandis que l'autre se charge de réaliser un histogramme permanent des températures acquises.

A ce niveau, il n'est pas nécessaire de comprendre toutes les structures du diagramme. Nous reviendrons abondamment sur chacun des éléments dans les chapitres suivants de ce tutorial.

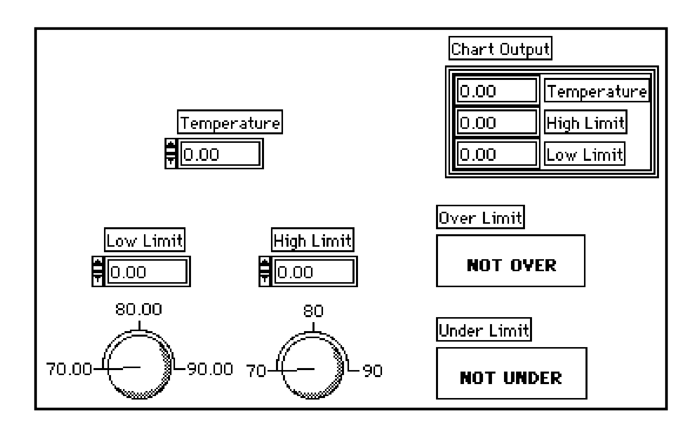
La hiérarchie

La puissance de LabVIEW réside dans la nature hiérarchique des VIs. En effet, une fois que vous avez créé un VI, vous pouvez ensuite l'utiliser comme *sous-VI* dans le diagramme d'un VI de niveau supérieur. Vous pouvez en outre concevoir des hiérarchies sur un nombre illimité de niveaux.

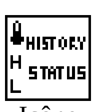


A titre d'exemple, observez un VI que le VI **Temperature System Demo** utilise comme sous-VI dans son diagramme.

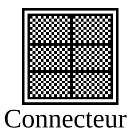
1. Ouvrez le sous-VI **Temperature Status** en double-cliquant sur son icône. Vous faites ainsi apparaître la face-avant suivante.



Le cadre icône/connecteur



Icône


Connecteur

Le cadre icône/connecteur permet de transformer un VI en un objet utilisable en tant que fonction ou sous-programme dans les diagrammes d'autres VIs. L'icône et le connecteur se trouvent en haut à droite de la face-avant d'un VI. L'icône correspond à une représentation graphique du VI dans le diagramme d'autres VIs. Les broches du connecteur correspondent aux points de connexion des entrées et des sorties de l'icône. Les *broches* s'apparentent aux paramètres d'un sous-programme ou d'une fonction. Elles correspondent aux commandes et aux indicateurs situés sur la face-avant du VI. L'icône masque le connecteur jusqu'à ce que vous le rendiez visible.

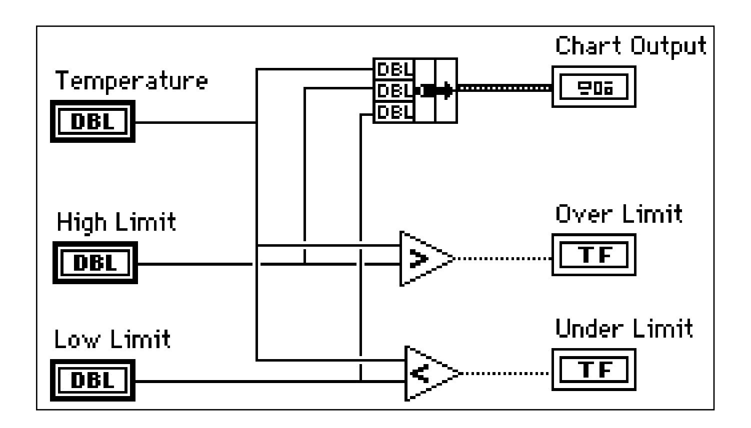
2. Placez l'outil Doigt sur le cadre icône en haut à droite de la face-avant et maintenez le bouton droit de la souris enfoncé. Vous voyez alors apparaître un menu local.

3. Sélectionnez **Show Connector** dans le menu local. Le curseur prend alors l'aspect de l'outil Bobine, représenté sur la gauche.



Les carrés figurant sur le connecteur représentent les broches qui correspondent elles-mêmes aux commandes et aux indicateurs de la face-avant.

4. Cliquez sur une broche, elle s'assombrit. Vous remarquerez qu'une commande ou qu'un indicateur est alors mis en évidence dans la face-avant. Lorsque vous câblez la commande ou l'indicateur (broche), les données qu'il/elle contient passent à (ou sont récupérées depuis) l'autre extrémité du fil.
5. Positionnez l'outil Bobine sur le connecteur de la face-avant et ouvrez un menu local. Il s'affiche alors à l'écran.
6. Sélectionnez **Show Icon**. L'outil Bobine reprend alors l'aspect de l'outil Doigt.
7. Revenez au diagramme en sélectionnant **Windows»Show Diagram.** A ce stade du tutorial, il n'est pas nécessaire de maîtriser le fonctionnement du diagramme. Retenez tout simplement qu'un sous-VI peut être en soi complexe ou à l'inverse très simple.

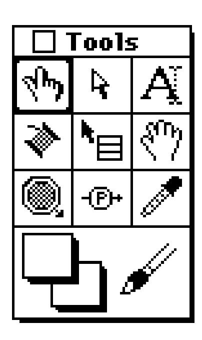


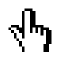
En créant des sous-VIs, vous pouvez rendre modulaires les diagrammes. Cette modularité facilite la mise au point, la compréhension et la maintenance des VIs.

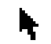
8. Revenez à la face-avant (**Windows»Show Panel**).
9. Sélectionnez **File»Close** et n'enregistrez aucune modification.

La palette Tools

LabVIEW utilise une palette **Tools** flottante, que vous pouvez utiliser pour modifier et mettre au point les VIs. Utilisez la touche `<Tab>` pour passer d'un outil à un autre parmi les plus souvent utilisés. Si vous avez fermé la palette **Tools**, sélectionnez **Windows»Show Tools Palette** pour l'afficher. L'illustration suivante représente la palette **Tools**.

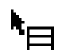


 outil Doigt : permet de positionner des éléments des palettes **Controls** et **Functions** sur la face-avant et dans le diagramme

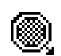
 outil Flèche : permet de positionner, redimensionner et sélectionner les objets

 outil Texte : permet de modifier du texte et d'en créer

 outil Bobine : permet de câbler des objets entre eux dans le diagramme

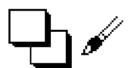
 outil Menu local : fait apparaître un menu local pour un objet

 outil Main : fait défiler toute la fenêtre sans avoir recours aux barres de défilement

 outil Point d'arrêt : permet de définir des points d'arrêt dans les VIs, les fonctions, les boucles, les séquences et les structures

 outil Sonde : permet de créer des sondes sur les fils

 outil Pipette : copie les couleurs pour les coller à l'aide de l'outil Pinceau



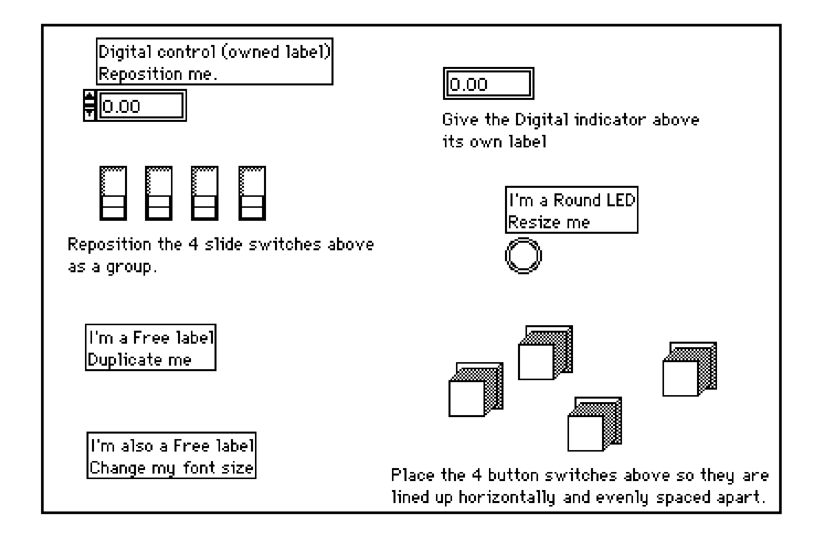
outil Pinceau : permet de définir la couleur de l'arrière-plan et du premier plan

Les techniques d'édition

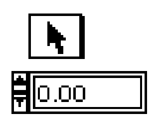
OBJECTIF Maîtriser les techniques d'édition de LabVIEW.

Pour travailler sur le VI **Editing Exercise**, sélectionnez **File»Open**. Le VI en question se trouve dans le répertoire `examples\general\controls\smplctls.llb.`

La face-avant du VI **Editing Exercise** contient un certain nombre d'objets LabVIEW. Votre objectif est de modifier cette face-avant afin qu'elle ressemble à celle de l'illustration suivante.



1. Si la palette **Tools** n'est pas visible, sélectionnez **Windows»Show Tools Palette** pour la rendre visible.
2. Déplacez la commande numérique.



   a. Choisissez l'outil Flèche dans la palette **Tools**.

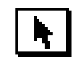
   b. Cliquez sur la commande numérique et faites-la glisser jusqu'à un nouvel emplacement.

Vous remarquerez que l'étiquette suit la commande au cours de son déplacement. En effet, l'étiquette est *dépendante* de la commande qu'elle nomme.

3. Cliquez ensuite dans une zone libre de la face-avant afin de désactiver la commande, puis cliquez sur l'étiquette et faites-la glisser jusqu'à un nouvel emplacement.

Cette fois-ci, vous remarquerez que la commande ne suit pas. En résumé, le déplacement des commandes entraîne celui de leur étiquette, alors que le déplacement des étiquettes n'entraîne pas celui des commandes correspondantes.

4. Passez au diagramme en sélectionnant **Windows»Show Diagram.**
5. Déplacez le diagramme de manière à visualiser les deux fenêtres.
6. Cliquez sur la face-avant pour l'activer.
7. Copiez la commande numérique pour créer une constante correspondante.



   a. Choisissez l'outil Flèche dans la palette **Tools**.



   b. Cliquez sur la commande numérique. Tout en maintenant le bouton de la souris enfoncé, faites-la glisser dans le diagramme. La commande numérique prend maintenant l'aspect d'une constante dans le diagramme. Vous pouvez également utiliser les options **Copy** et **Paste** du menu **Edit** pour copier la commande et la dupliquer dans le diagramme.

☞ **Remarque :** ***vous pouvez procéder de la même façon pour faire glisser ou copier une constante du diagramme vers la face-avant pour créer une commande correspondante.***

8. Déplacez simultanément les quatre interrupteurs à glissière.



   a. Avec l'outil Flèche, cliquez dans une zone libre proche des quatre interrupteurs, maintenez enfoncé le bouton de la souris et faites glisser le pointeur jusqu'à ce que les interrupteurs se trouvent tous à l'intérieur du même rectangle de sélection.
   b. Cliquez sur le groupe d'interrupteurs et déplacez-le.

9. Dupliquez l'étiquette libre. Avec l'outil Flèche, maintenez enfoncée la touche `<Ctrl>` (**Windows**) ; `<option>` (**Macintosh**) ; `<meta>` (**Sun**) ; ou `<Alt>` (**HP-UX**), cliquez sur l'étiquette libre, puis faites glisser le duplicata sur une autre zone. Sous UNIX, vous pouvez utiliser le bouton central de la souris



pour faire glisser l'étiquette. Vous créez ainsi un duplicata de l'étiquette.

10. Modifiez la taille des caractères de l'étiquette.



   a. Sélectionnez l'étiquette en utilisant l'outil Texte. Vous pouvez soit cliquer trois fois de suite sur le texte, soit cliquer sur la première lettre, puis faire glisser le pointeur à la fin de celui-ci.

   b. Modifiez la taille des caractères du texte sélectionné en 12 points en choisissant **Size** à partir du menu **Police de caractères** dans la barre d'outils.

9pt Application Font

11. Créez une étiquette pour l'indicateur numérique.

   a. Ouvrez un menu local sur l'indicateur numérique et choisissez **Show»Label**.



   b. Tapez Indicateur numérique dans la boîte encadrée, puis cliquez avec le bouton de la souris en dehors de l'étiquette. Vous pouvez indiquer que vous avez terminé de saisir du texte en appuyant sur la touche `<Enter>` du clavier numérique.



12. Redimensionnez le voyant lumineux rond. Placez l'outil Flèche sur un coin du voyant jusqu'à ce que la flèche se transforme en curseur de redimensionnement. Cliquez et faites glisser le curseur pour agrandir le voyant lumineux. Si vous souhaitez conserver les proportions (rapport hauteur/largeur) du voyant, maintenez enfoncée la touche `<Maj>` pendant l'opération de redimensionnement.



13. Modifiez la couleur du voyant lumineux rond.



   a. Avec l'outil Pinceau, ouvrez un menu local sur le voyant lumineux.

   b. Choisissez une couleur dans la palette de sélection. Le fait de relâcher le bouton de la souris valide la dernière couleur sélectionnée.

14. Alignez les quatre boutons-poussoirs et espacez-les à intervalle régulier.

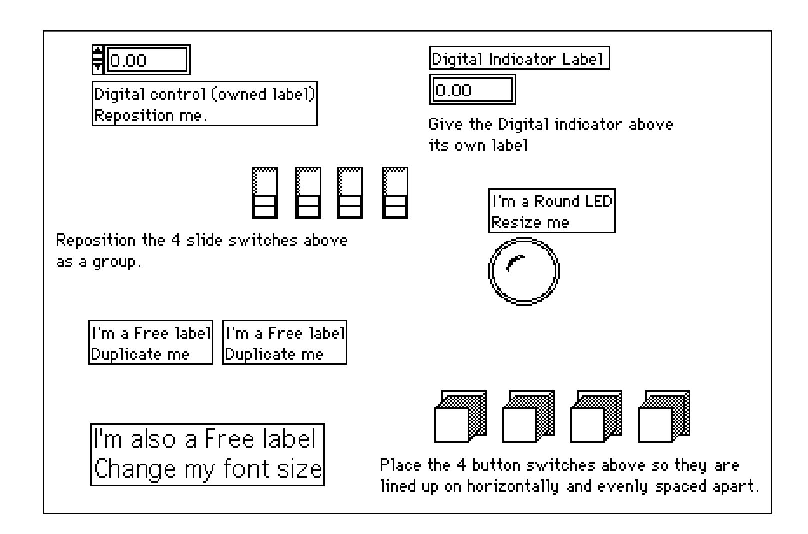
   a. Avec l'outil Flèche, cliquez dans une zone libre proche des quatre boutons et faites glisser le pointeur jusqu'à ce que tous les boutons se trouvent à l'intérieur du rectangle de sélection.



   b. Alignez les boutons en cliquant sur le menu **Alignement** dans la barre d'outils et en choisissant l'option **Vertical Centers**.



c. Espacez les boutons à intervalle régulier en cliquant sur le menu **Distribution** et en choisissant l'option **Horizontal Centers**. Après toutes ces opérations, votre face-avant devrait ressembler à celle de l'illustration suivante.

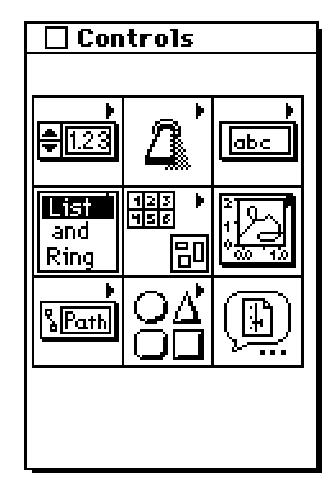


15. Fermez le VI en choisissant **File»Close**. N'enregistrez aucune modification.

La palette Controls

La palette **Controls** est une palette graphique flottante qui s'ouvre automatiquement dès que vous lancez LabVIEW. Elle vous permet de placer les commandes et les indicateurs dans la face-avant d'un VI. Chaque icône principale contient des sous-palettes. Si la palette **Controls** n'est pas affichée, vous pouvez la rendre visible en sélectionnant **Windows»Show Controls Palette** dans le menu de la face-avant. Vous pouvez également ouvrir un menu local dans une zone libre de la face-avant pour accéder à une copie temporaire de la palette **Controls**.

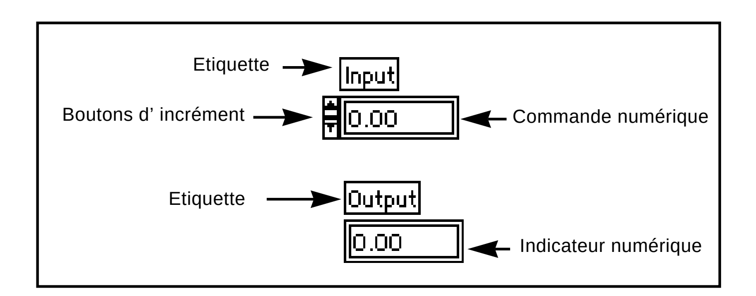
L'illustration suivante présente le niveau principal de la palette **Controls**.



Commandes et indicateurs

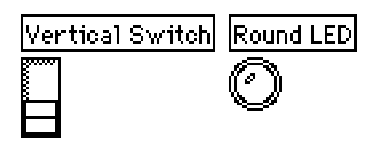
Commandes et indicateurs numériques

Les commandes numériques servent à entrer des quantités numériques, alors que les indicateurs numériques affichent d'autres quantités numériques. Les deux objets les plus souvent utilisés sont les *commandes numériques* et les *indicateurs numériques*.



Commandes et indicateurs booléens

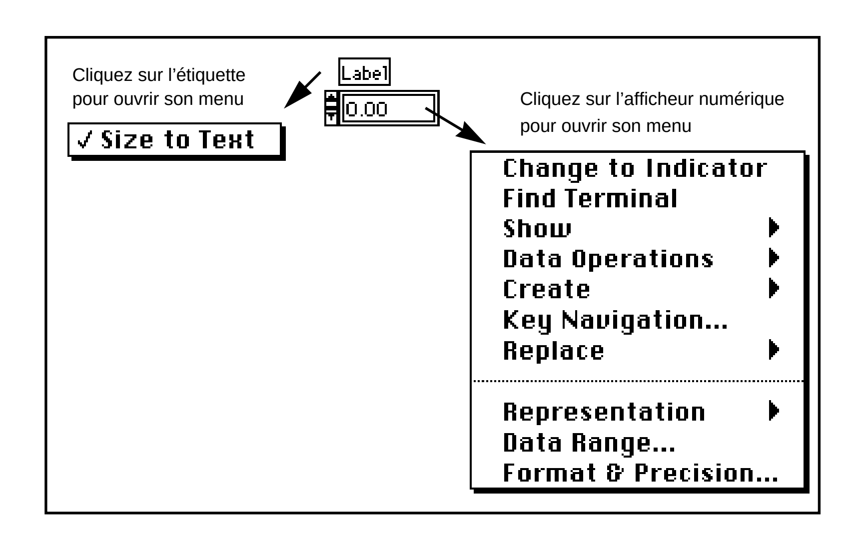
Les commandes et les indicateurs booléens servent à saisir et à afficher les valeurs booléennes (TRUE/FALSE). Les objets booléens simulent des interrupteurs, des boutons et des voyants lumineux. Les objets booléens les plus souvent utilisés sont les *interrupteurs verticaux* et les *voyants lumineux ronds*.



Configuration des commandes et des indicateurs

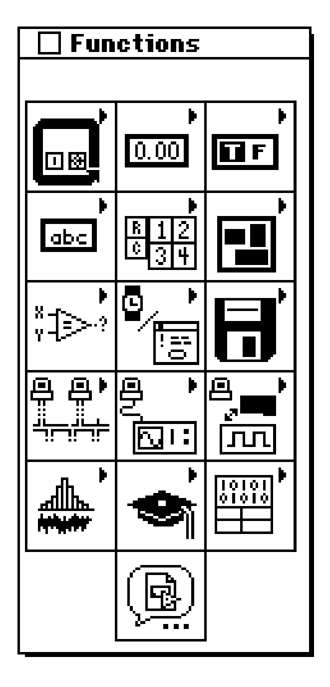


Vous pouvez configurer pratiquement tous les indicateurs et commandes en utilisant les options de leurs menus locaux respectifs. *Le seul fait d'ouvrir un menu local sur les éléments des commandes et des indicateurs permet d'afficher des menus servant à les personnaliser*. La façon la plus simple d'accéder au menu local consiste à cliquer sur l'outil Menu local, présenté sur la gauche, sur tous les objets dotés d'un menu local. La figure suivante illustre cette méthode d'affichage pour une commande numérique.



La palette Functions

La palette **Functions** est une palette graphique flottante qui s'ouvre automatiquement dès que vous passez dans le diagramme d'un VI. Cette palette sert à placer des nœuds (constantes, indicateurs, VIs, etc.) dans le diagramme d'un VI. Toutes les icônes principales contiennent des sous-palettes. Si la palette **Functions** n'est pas affichée, sélectionnez **Windows»Show Functions Palette** dans le menu du diagramme pour la rendre visible. Vous pouvez également ouvrir un menu local sur une zone libre du diagramme pour accéder à une copie temporaire de la palette **Functions**. La figure suivante est une représentation du niveau principal de la palette **Functions**.



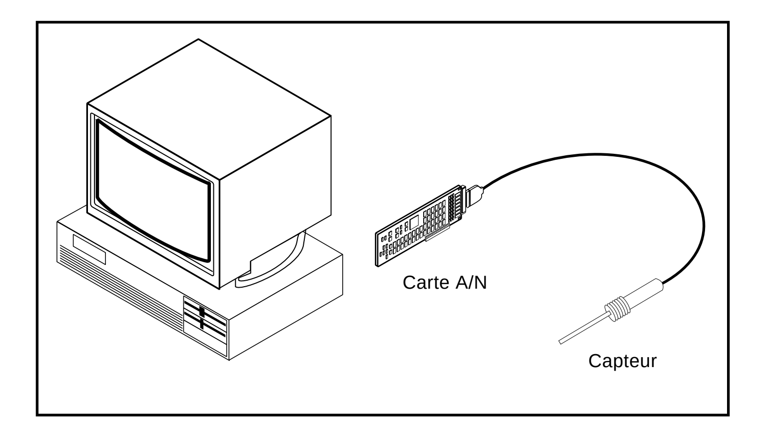
La construction d'un VI

OBJECTIF Construire un VI qui simule l'acquisition de mesure de température.

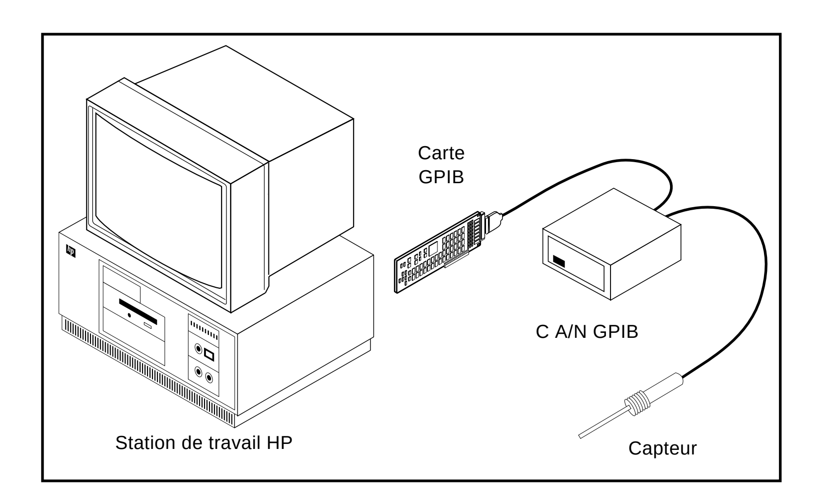
Vous utiliserez le VI **Demo Voltage Read** pour mesurer la tension, puis multiplierez le résultat par 100.0 pour convertir la tension en une température exprimée en degrés (Fahrenheit).

Imaginez que vous disposez d'un transducteur ou d'un capteur qui convertit la température en tension.

(Windows, Macintosh et Sun) Ce capteur est connecté à une carte de convertisseur analogique-numérique (C A/N), comme dans l'exemple reproduit ici, qui convertit la tension en données numériques.



(HP-UX) Le capteur pourrait aussi bien être connecté à un convertisseur analogique-numérique connecté à l'ordinateur via une interface GPIB, comme dans l'illustration suivante. Cette méthode permet aussi de convertir la tension en données numériques.

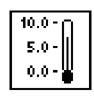


La face-avant

1. Pour ouvrir une nouvelle face-avant, sélectionnez **File»New**. Pour les plates-formes Windows et UNIX, si vous avez fermé tous les VIs, choisissez alors **New VI** dans la fenêtre de dialogue LabVIEW.



☞ **Remarque :** ***si la palette*** **Controls** ***n'est pas affichée, choisissez*** **Windows»Show Controls Palette** ***pour la rendre visible. Vous pouvez également accéder à la palette*** **Controls** ***en ouvrant un menu local dans une zone libre de la face-avant.***

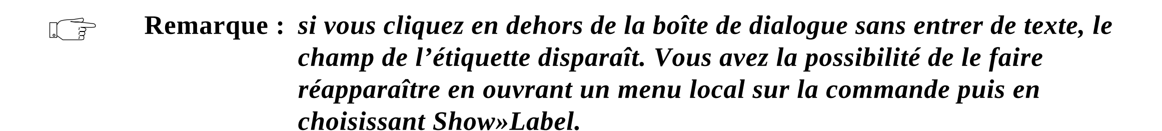




2. Sélectionnez un indicateur **Thermometer** dans **Controls»Numeric**, puis placez-le dans la face-avant.

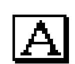


3. Tapez `Temp` à l'intérieur de l'étiquette puis cliquez sur le bouton Enter de la barre d'outils.



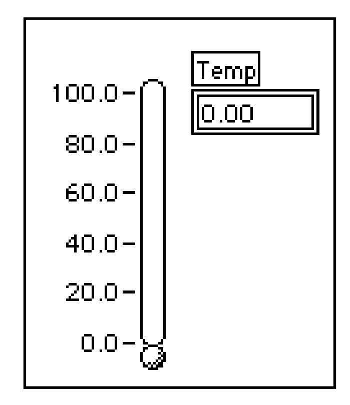
☞ **Remarque :** ***si vous cliquez en dehors de la boîte de dialogue sans entrer de texte, le champ de l'étiquette disparaît. Vous avez la possibilité de le faire réapparaître en ouvrant un menu local sur la commande puis en choisissant Show»Label.***

N'oubliez pas que pour ouvrir un menu local, vous devez cliquer sur le bouton droit de la souris (`<command>` ***-click pour Macintosh).***

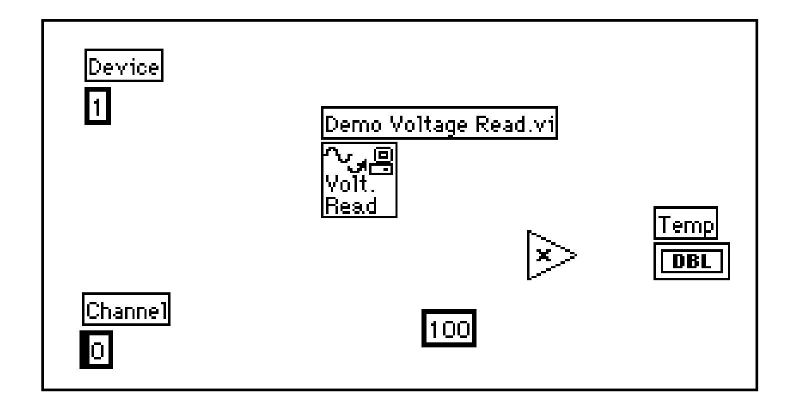


4. Modifiez l'échelle du thermomètre pour afficher la température entre 0.0 et 100.0.
   a. Pour cela, avec l'outil Texte, double-cliquez sur `10.0` dans l'échelle du thermomètre pour mettre cette valeur en évidence.
   b. Tapez `100.0` dans l'échelle puis cliquez avec le bouton de la souris n'importe où en dehors de la fenêtre d'affichage. LabVIEW met automatiquement à l'échelle les valeurs intermédiaires. Le thermomètre devrait maintenant ressembler à celui de l'illustration suivante.

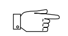




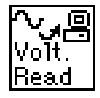
Le diagramme



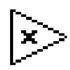
1. Pour ouvrir le diagramme, choisissez les options **Windows»Show Diagram**. Sélectionnez les objets qui vont composer le diagramme dans la palette **Functions**. Pour chaque objet à insérer, sélectionnez l'icône puis l'objet dans le niveau principal de la palette, ou choisissez l'objet dans la sous-palette appropriée. Lorsque vous placez le pointeur de la souris dans le diagramme, LabVIEW affiche les contours de l'objet. C'est à l'utilisateur de choisir l'endroit où l'objet sera placé dans le diagramme.



Remarque : ***si la palette* Functions *n'est pas affichée, sélectionnez* Windows»Show Functions Palette *pour la rendre visible. Vous pouvez également accéder à la palette* Functions *en ouvrant un menu local dans une zone libre du diagramme.***



Le VI **Demo Voltage Read** (**Functions»Tutorial)** simule la mesure d'une tension acquise par une carte d'acquisition de données.

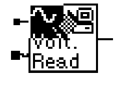


Fonction **Multiply** (**Functions»Numeric)**. Dans cet exercice, la fonction multiplie par `100.0` la tension donnée par le VI **Demo Voltage Read**.



Numeric constant (**Functions»Numeric)**. Deux constantes numériques sont nécessaires : une pour le facteur d'échelle 100 et une autre pour la constante du périphérique. Pour la première constante numérique, tapez `100.0` dès que la constante s'affiche dans le diagramme.

2. Créez la seconde constante numérique en utilisant une touche de raccourci. Vous créez et câblez ainsi automatiquement la constante au VI **Demo Voltage Read**.





   a. A l'aide de l'outil Bobine, ouvrez un menu local sur l'entrée `Board ID` du VI **Demo Voltage Read** puis sélectionnez **Create Constant** dans le menu local. Cette option permet de créer automatiquement une constante numérique et de la câbler au VI **Demo Voltage Read**.

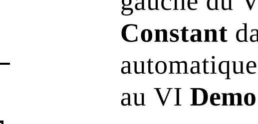
   b. Tapez `1` lorsque la constante apparaît pour la première fois dans le diagramme. Cette manipulation permet de modifier la valeur par défaut qui passe de 0 à 1. Vous remarquerez qu'il n'est pas nécessaire de modifier l'outil Texte pour insérer la valeur en procédant de cette manière.

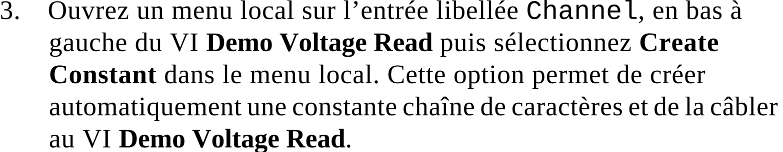
   c. A l'aide de l'outil Texte, modifiez le texte de l'étiquette active (Board ID) en `Device.`

Dans cet exemple, les deux nombres représentent la constante 100.0 et le périphérique pour la fonction **Multiply**.



String Constant (**Functions»String).**





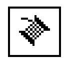
3. Ouvrez un menu local sur l'entrée libellée `Channel`, en bas à gauche du VI **Demo Voltage Read** puis sélectionnez **Create Constant** dans le menu local. Cette option permet de créer automatiquement une constante chaîne de caractères et de la câbler au VI **Demo Voltage Read**.



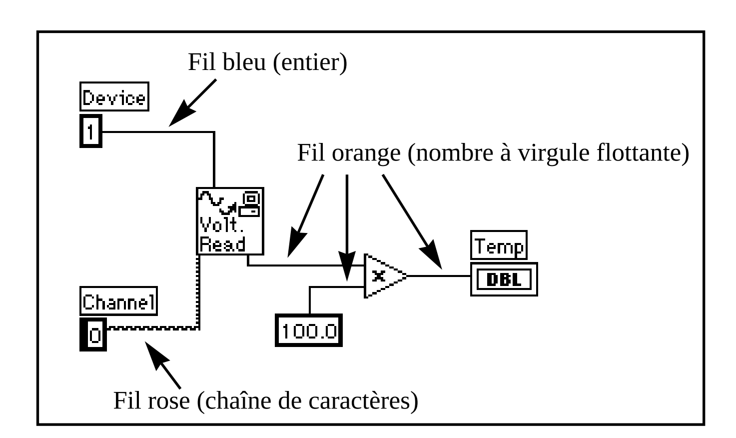
4. Tapez `0` dès que la constante apparaît. Dans cet exemple, vous remarquerez que le terme `Channel` s'affiche dans l'étiquette active si bien qu'il n'est pas nécessaire de modifier l'étiquette.

Dans cet exemple, vous utilisez une constante chaîne de caractères pour indiquer le numéro du canal.

☞ **Remarque :** ***vous pouvez créer et câbler des commandes, des constantes et des indicateurs avec la plupart des fonctions. Si ces options ne sont pas disponibles pour une fonction en particulier, les options* Create Control, Create Constant *et* Create Indicator *sont désactivées dans le menu local. Pour plus de détail sur cette caractéristique, veuillez vous reporter à la section*** *Création et câblage des commandes, constantes et indicateurs* ***ultérieurement dans ce chapitre.***



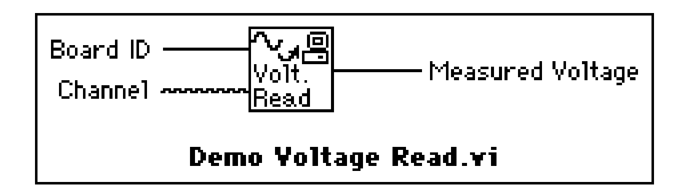
5. A l'aide de l'outil Bobine, câblez les objets restants conformément aux explications données dans la section intitulée *Techniques de câblage*, plus loin dans ce chapitre.



Toutes les connexions LabVIEW sont identifiées par une couleur, laquelle est fonction du type de données acheminées. Ainsi les fils bleus véhiculent des nombres entiers, les fils orange des nombres à virgule flottante, les fils verts des nombres booléens et les fils roses des chaînes de caractères.

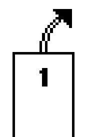
Vous pouvez activer la fenêtre d'aide en choisissant **Help»Show Help**. En plaçant n'importe quel outil d'édition sur un nœud, vous visualisez les entrées et les sorties de cette fonction dans la fenêtre d'aide. Au moment où vous passez l'outil d'édition sur l'icône du VI, LabVIEW met en

surbrillance les broches de terminaison dans les fenêtres du diagramme et d’aide. Lorsque vous commencerez à câbler vos diagrammes, cette mise en évidence vous aidera à connecter les entrées et les sorties aux bons terminaux.



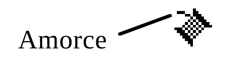
Le VI **Demo Voltage Read** simule la mesure d’une tension acquise sur le canal 0 d’une carte d’acquisition enfichable. Le VI multiplie alors la tension par 100.0 pour la convertir en une température exprimée en degrés Fahrenheit.

Les techniques de câblage



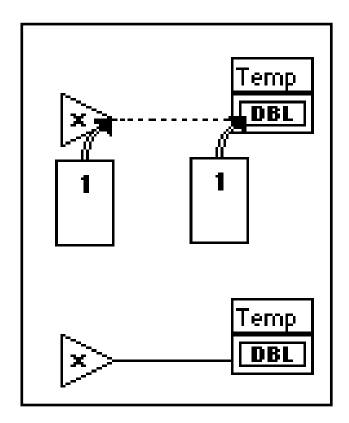
Dans les exemples de câblage reproduits dans cette section, la flèche située à l’extrémité du symbole de la souris indique l’endroit où cliquer et le chiffre figurant sur le bouton de la souris indique le nombre de fois qu’il faut cliquer.

L’*amorce* (ou *point actif*) de l’outil est l’extrémité du morceau de fil dévidé.

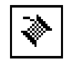


Pour connecter un terminal à un autre, cliquez avec l’outil Bobine sur le premier terminal, déplacez la bobine sur le second terminal, puis cliquez dessus. Vous pouvez commencer par n’importe quel terminal, cela n’a aucune importance.

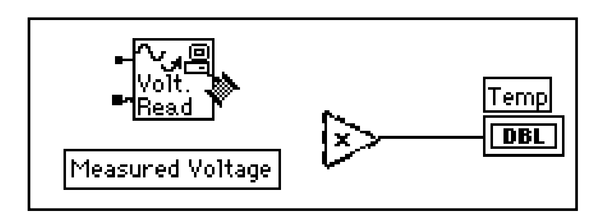
Lorsque l’outil Bobine se trouve au-dessus d’un terminal, la zone concernée clignote, signalant ainsi que vous pouvez cliquer pour établir la connexion. Surtout, ne maintenez *pas* le bouton de la souris enfoncé lorsque vous déplacez l’outil Bobine d’un terminal à un autre. Vous pouvez infléchir la direction d’un fil une seule fois en faisant glisser la souris perpendiculairement à la direction initiale. Pour infléchir la direction du fil plusieurs fois, cliquez sur le bouton de la souris. Pour modifier l’orientation du fil, appuyez sur la barre d’espace. Cliquez sur le bouton de la souris pour *maintenir en place* le fil et faites glisser la souris perpendiculairement.



Info-bulles



Lorsque vous placez l'outil Bobine au-dessus du terminal d'un nœud, une info-bulle apparaît. Il s'agit de fanions jaunes qui reprennent le nom de chaque terminal. Ces info-bulles vous aideront à câbler les terminaux. La figure suivante illustre l'info-bulle (**Measured Voltage**) qui apparaît lorsque vous placez l'outil Bobine au-dessus de la sortie du VI **Demo Voltage Read**.



Remarque : ***lorsque vous placez l'outil Bobine sur un nœud, LabVIEW affiche des mauvaises connexions pour signaler chaque entrée et chaque sortie. Une mauvaise connexion avec un point à son extrémité représente l'entrée à un nœud.***

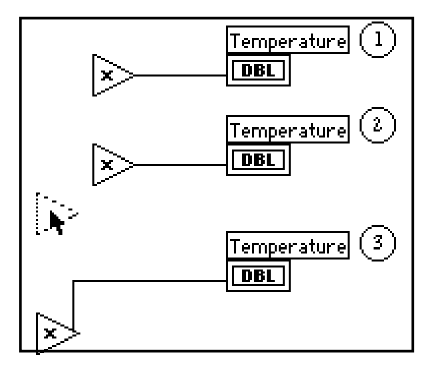
Visualisation des terminaux

Il est important que, pour chaque fonction, les connexions soient effectuées aux bons endroits. Pour vous faciliter le travail, vous pouvez visualiser le connecteur de chaque icône. Pour ce faire, ouvrez le menu local correspondant à la fonction et choisissez **Show Terminals**. Pour revenir à l'icône, ouvrez le menu local de la fonction et sélectionnez de nouveau **Show Terminals**.

Adaptation des liaisons

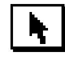


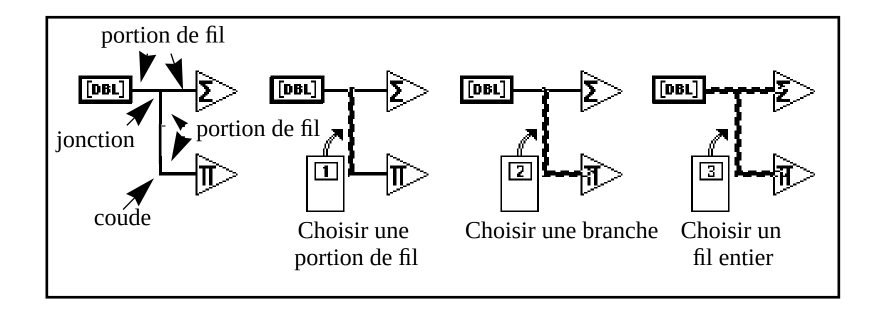
Vous pouvez déplacer les objets câblés soit individuellement soit en groupe en faisant glisser les objets sélectionnés avec l'outil Flèche.



Sélection et suppression des liaisons

Il peut vous arriver de faire une erreur dans le câblage des nœuds. Le cas échéant, sélectionnez le fil à supprimer puis appuyez sur la touche `<Suppr>`. Une *portion* de fil désigne la partie horizontale ou verticale d'une connexion. Le point où se rejoignent les trois ou quatre portions de fil s'appelle une jonction. Une *branche* rassemble toutes les portions de fil entre les jonctions, d'un terminal à la prochaine jonction, ou d'un terminal à un autre s'il n'existe aucune jonction entre les deux. Pour sélectionner une portion de fil, cliquez dessus avec l'outil Flèche. Un double clic sélectionne une branche entière, un triple clic sélectionne la totalité du fil.



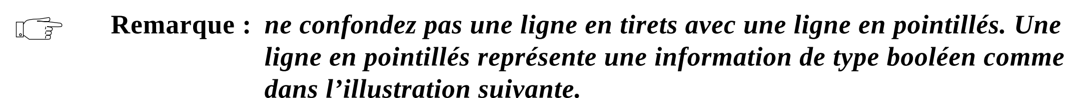


Mauvaises liaisons

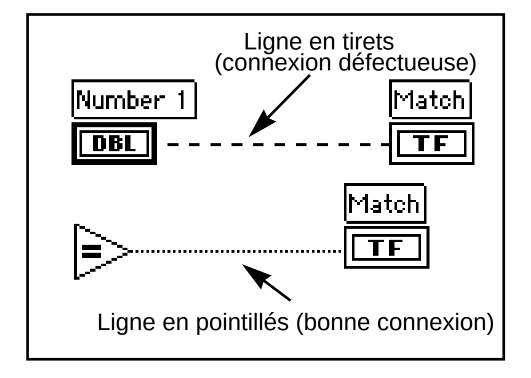
Une ligne constituée de tirets représente une connexion défectueuse qui peut avoir plusieurs origines. Il peut s'agir d'une connexion entre deux organes de commande, ou d'une connexion entre un terminal source et un terminal de destination alors que les données sont incompatibles (par exemple, des données numériques et des données booléennes). Vous pouvez supprimer une mauvaise connexion en cliquant dessus à l'aide de l'outil Flèche puis en appuyant sur la touche



`<Suppr>`. En choisissant les options **Edit»Remove Bad Wires**, vous supprimez toutes les connexions défectueuses du diagramme. Vous vous en servirez si votre VI refuse de fonctionner correctement ou s'il renvoie le message d'erreur `Signal has loose ends`.

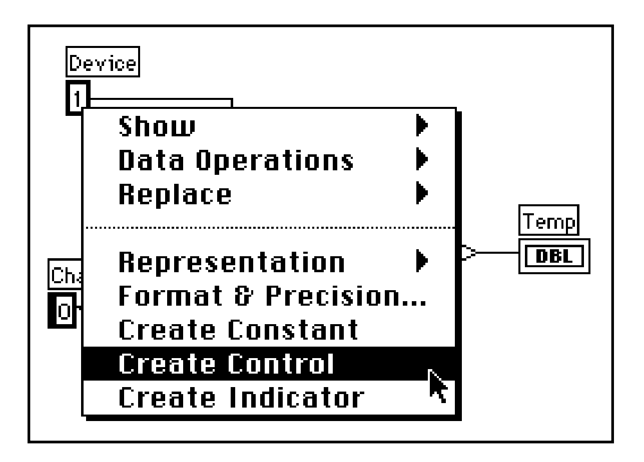


Remarque : ***ne confondez pas une ligne en tirets avec une ligne en pointillés. Une ligne en pointillés représente une information de type booléen comme dans l'illustration suivante.***



Création et câblage des commandes, constantes et indicateurs

Pour les terminaux fonctionnant comme des entrées dans le diagramme, LabVIEW offre deux fonctionnalités que vous pouvez utiliser pour créer et câbler une commande ou une constante. Vous accédez à ces fonctionnalités en ouvrant un menu local sur le terminal et en choisissant **Create Control** ou **Create Constant**. LabVIEW crée et câble automatiquement le bon type de commande ou de constante à l'entrée du terminal. L'illustration suivante est un exemple de menu local.



Pour un terminal fonctionnant comme une sortie dans le diagramme, vous pouvez choisir la fonctionnalité **Create Indicator** pour créer puis câbler un indicateur au terminal. Vous accédez à cette fonctionnalité en ouvrant un menu local sur le terminal et en choisissant **Create Indicator**. LabVIEW crée et câble automatiquement le bon type d'indicateur à la sortie du terminal.

Exécution du VI

1. Pour les plates-formes Windows et Macintosh, activez la face-avant en cliquant n'importe où sur celle-ci. Sous UNIX, activez la face-avant en cliquant sur la barre de titre de la fenêtre ou en choisissant les options **Windows»Show Panel**.
2. Lancez le VI en cliquant sur le bouton Exécution de la barre d'outils de la face-avant.



Vous remarquerez que vous devez relancer le VI à chaque fois. Si vous souhaitez une exécution permanente, vous devez cliquer sur le bouton Exécution permanente.



3. Cliquez sur le bouton Exécution permanente de la barre d'outils.
4. Cliquez une deuxième fois sur le bouton Exécution permanente pour le désactiver. Le VI termine alors l'exécution et se ferme.

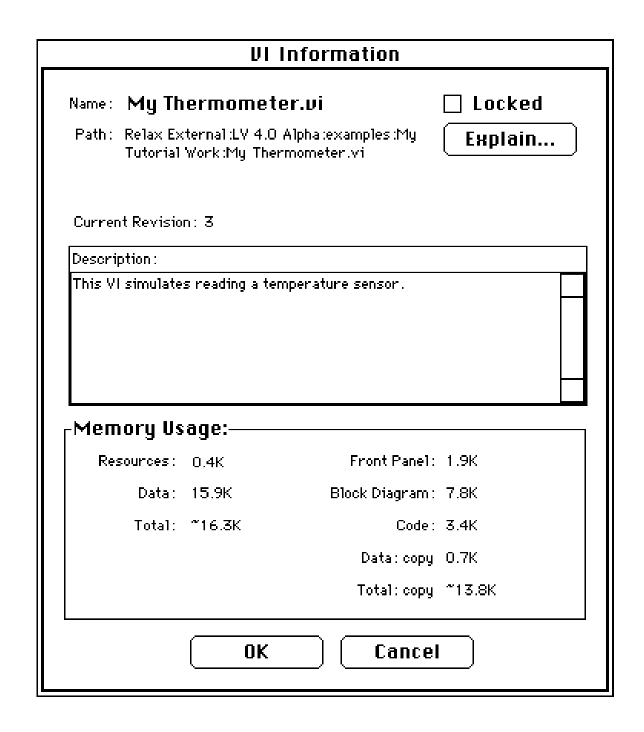


Remarque : ***le bouton Exécution permanente n'est pas le meilleur moyen pour renouveler indéfiniment l'exécution du code du diagramme. Il est préférable d'utiliser une structure de bouclage. Cet aspect est abordé dans le chapitre 3,*** *Boucles et graphes déroulants****, de ce tutorial.***

Documentation d'un VI

Vous pouvez documenter un VI en choisissant **Windows»Show VI Info...**. Tapez le texte de la description du VI dans la boîte de dialogue **Informations VI**. Vous pouvez alors rappeler ce texte en choisissant à nouveau **Windows»Show VI Info...**.

1. Documentez le VI. Sélectionnez **Windows»Show VI Info...**. Tapez le texte de la description du VI, conformément à l'exemple suivant, puis cliquez sur **OK**.



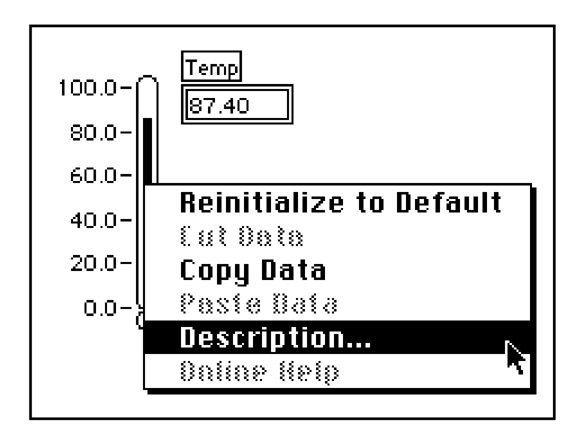
Vous avez la possibilité de visualiser la description des objets sur la face-avant (ou leurs terminaux respectifs dans le diagramme) en ouvrant un menu local sur l'objet et en choisissant **Description...**. L'emplacement de ce choix varie selon qu'il s'agit de la face-avant ou d'un diagramme.

face-avant : ouvrez un menu local sur l'objet et choisissez **Data Operations»Description...**.

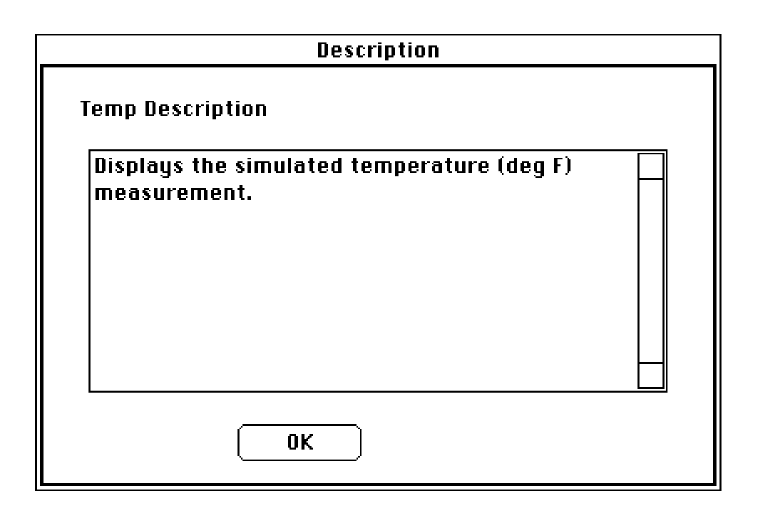
diagramme : ouvrez un menu local sur l'objet et choisissez **Description...**.

☞ **Remarque :** ***il est impossible de modifier le texte de la description tout en exécutant un VI.***

L’illustration suivante est un exemple de menu local qui s’affiche pendant que vous exécutez un VI. Vous ne pouvez ajouter ni modifier le texte de la description pendant l’exécution du VI. En revanche, vous pouvez visualiser toutes les informations préalablement enregistrées.



2. Documentez l’indicateur thermomètre.
   a. Dans la face-avant, ouvrez un menu local sur l’indicateur thermomètre et choisissez **Data Operations»Description...**.
   b. Tapez le texte de la description du thermomètre, conformément à l’exemple suivant, puis cliquez sur **OK**.



3. Affichez le texte de la description que vous venez de créer en ouvrant un menu local sur l'indicateur thermomètre et en choisissant **Data Operations»Description...**.

Enregistrement et chargement des VIs

A l'instar de toutes les autres applications, vous pouvez enregistrer votre VI dans un fichier ou un répertoire traditionnel. Avec LabVIEW, vous pouvez également enregistrer plusieurs VIs dans un fichier unique appelé *bibliothèque de VIs*. La bibliothèque `tutorial.llb` en est un exemple.

Si vous utilisez Windows 3.1, nous vous recommandons d'enregistrer vos VIs dans les bibliothèques de VIs car vous pouvez utiliser des noms de fichier assez longs, (jusqu'à 255 caractères) en majuscules et en minuscules.

Nous vous déconseillons d'utiliser les bibliothèques de VIs sauf si vous devez transférer vos VIs dans Windows 3.1. Enregistrer les VIs sous la forme de fichiers individuels est une opération plus rentable car vous avez ainsi la possibilité de copier, renommer et supprimer plus facilement les fichiers que si vous utilisiez une bibliothèque de VIs. Pour connaître les avantages et les inconvénients des bibliothèques de VIs et des fichiers séparés, veuillez vous reporter à la section *Enregistrement des VIs* du chapitre 2, *La création des VIs*, du *Manuel de l'utilisateur LabVIEW*.

Cela étant, il est bon que vous sachiez comment ces bibliothèques fonctionnent. Par conséquent, nous vous invitons à enregistrer tous les VIs que vous allez créer au cours de cette initiation dans des bibliothèques de VIs.

Enregistrez votre VI dans une bibliothèque de VIs.

1. Sélectionnez **File»Save As...**. Sous UNIX, déplacez-vous dans le système de fichier là où vous avez des privilèges d'écriture. Par exemple, vous pouvez choisir votre répertoire personnel.
2. N'enregistrez pas vos fichiers dans le répertoire `examples`. A la place, créez votre propre répertoire et nommez-le `Tutorial VIs`.

3. Créez la bibliothèque de VIs.

   (Windows) Sélectionnez **New...** ou le bouton **New VI Library** pour créer la bibliothèque de VIs.

   (Macintosh) Si vous utilisez la boîte de dialogue fichier native, **Edit»Preferences...**, sélectionnez **Use LLBs** pour avoir accès à la boîte de dialogue fichier de LabVIEW. Pour créer la bibliothèque de VIs, cliquez sur **Save»New...**.

   (UNIX) Sélectionnez **Save»New...**.
4. Nommez `mywork` la nouvelle bibliothèque dans la boîte de dialogue puis cliquez sur le bouton **VI Library**. Le nom de la bibliothèque doit être suivi de l'extension `.llb`. Pour Windows 3.1, vous devez limiter les noms de bibliothèques à huit caractères. LabVIEW ajoute automatiquement l'extension `.llb` si vous ne la mentionnez pas.

   Les bibliothèques de VIs présentent les mêmes caractéristiques de chargement, d'enregistrement et d'ouverture que les répertoires. Néanmoins, elles ne sont pas hiérarchiques : elles ne peuvent pas s'imbriquer les unes dans les autres. Vous ne pouvez pas non plus créer de nouveau répertoire à l'intérieur d'une bibliothèque de VIs. Sorti de l'environnement LabVIEW, vous n'avez aucun recours pour visualiser la liste des VIs contenus dans une bibliothèque VI.

Une fois que vous avez créé votre propre bibliothèque de VIs, elle apparaît dans la boîte de dialogue fichier de LabVIEW sous la forme d'un dossier avec la mention VI sur l'icône représentant le dossier. Les autres répertoires courants apparaissent sous la forme d'un dossier sans l'étiquette VI.

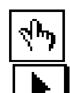
Répertoire
Bibliothéque de VIs
fichier
VI

5. Attribuez un nom au VI et enregistrez-le dans votre nouvelle bibliothèque. Vérifiez le nom qui figure dans la commande de type roue codeuse située en haut de la boîte de dialogue. Assurez-vous que le nom correspond bien à `mywork.llb`. Si tel n'est pas le cas, cliquez sur `mywork.llb` dans la liste de répertoires pour être sûr d'avoir bien enregistré votre VI au bon endroit.
    a. Tapez `My Thermometer.vi` dans la boîte de dialogue.
    b. Cliquez sur **OK**.
6. Fermez le VI en choisissant **File»Close.**

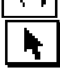
Résumé

Les instruments virtuels ou VIs sont composés de trois éléments principaux : la face-avant, le diagramme et l'icône/connecteur. La face-avant répertorie les entrées et les sorties du VI. Le diagramme équivaut au code exécutable du programme que vous créez en agençant des nœuds, des terminaux et des fils de liaison. Avec l'icône/connecteur, vous pouvez utiliser un VI comme sous-VI dans le diagramme d'un autre VI.

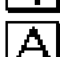
La palette **Tools** est une palette graphique flottante. Dans la face-avant et dans le diagramme, vous utilisez les outils de la palette **Tools** pour construire, éditer et mettre au point les VIs. Vous utilisez la touche <Tab> pour passer d'un outil à un autre parmi les plus utilisés de la palette. Les outils les plus fréquemment utilisés sont :



l'outil Doigt



l'outil Flèche



l'outil Texte



l'outil Bobine



l'outil Pinceau

Vous utilisez l'outil Doigt pour manipuler les commandes et les indicateurs de la face-avant. L'outil Flèche sert à placer, redimensionner et sélectionner les objets. L'outil Texte permet de créer des étiquettes libres et de saisir du texte dans les étiquettes. L'outil Bobine sert à connecter les objets entre eux dans le diagramme. Vous utiliserez l'outil Pinceau pour attribuer une couleur au premier plan et à l'arrière-plan des fenêtres, commandes, indicateurs, etc.

La face-avant et le diagramme contiennent des barres d’outils qui affichent le bouton Exécution ainsi que les autres boutons permettant de gérer l’exécution du VI.

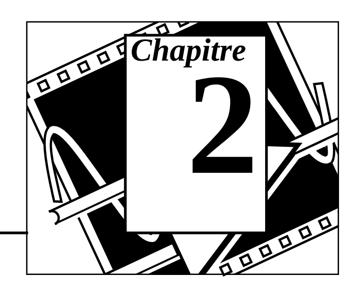
C’est dans la face-avant que vous placez des commandes et des indicateurs pour spécifier les entrées et les sorties du VI. Vous utilisez la palette **Controls** pour ajouter des commandes et des indicateurs dans la fenêtre de la face-avant. La palette **Controls** ouvre automatiquement un menu local sur la face-avant lorsque vous lancez LabVIEW. Vous pouvez également accéder à la palette **Controls** en choisissant **Windows»Show Controls Palette**. Les commandes et les indicateurs offrent de multiples options que vous configurez à partir de leurs menus locaux respectifs. Veuillez vous reporter au *Manuel de l’utilisateur LabVIEW* pour obtenir plus d’informations sur les commandes et les indicateurs de la face-avant.

Le diagramme contient des boutons, qui incluent également des caractéristiques que vous pouvez utiliser pour procéder à la mise au point des VIs et passer en mode exécution pas à pas.

Dans le diagramme, vous développez le diagramme source en connectant des nœuds et des terminaux à l’aide de l’outil Bobine. Vous utilisez la palette **Functions** pour placer les nœuds (structures, fonctions et sous-VIs) dans le diagramme. La palette **Functions** ouvre automatiquement un menu local sur le diagramme lorsque vous l’ouvrez. Vous pouvez également ouvrir la palette **Functions** en sélectionnant **Windows»Show Functions Palette**. LabVIEW place alors automatiquement sur le diagramme les terminaux associés aux commandes et aux indicateurs de la face-avant. Veuillez vous reporter au *Manuel de l’utilisateur LabVIEW* pour plus d’informations sur la programmation des diagrammes.

Vous pouvez pratiquement modifier tous les objets de LabVIEW avec les menus locaux. Pour accéder à ces menus, il vous suffit de cliquer sur l’objet ou d’utiliser l’outil Menu local.

Le fait d’ouvrir un menu local sur des parties séparées d’un objet vous permet d’accéder à leur propre menu local. Donc n’oubliez pas, si vous avez un doute, pensez au menu local !





La création d’un sous-VI

Vous allez apprendre :

- Ce qu’est un sous-VI.
- Comment créer son icône et son connecteur.
- Comment utiliser un VI en tant que sous-VI.

Le concept de hiérarchie

L’un des principes fondamentaux liés à la création d’applications LabVIEW réside dans la compréhension et la mise en œuvre de la nature hiérarchique des VIs. Après avoir créé le VI, vous pouvez l’utiliser comme *sous-VI* dans le diagramme d’un VI de niveau supérieur. Par conséquent, un sous-VI est comparable à un sous-programme en C. De même que le nombre de sous-programmes n’est pas limité pour la programmation en C, il ne l’est pas non plus pour les programmes développés avec LabVIEW. Vous pouvez également appeler un sous-VI dans un autre sous-VI.

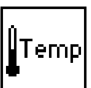
Lorsque vous créez une application, vous commencez par le VI principal, puis définissez les entrées et les sorties du système à développer. Vous construisez ensuite les sous-VIs pour mettre en œuvre les opérations sur les données circulant dans le diagramme. Si un diagramme contient un grand nombre d’icônes, regroupez-les dans un VI de niveau inférieur pour simplifier le diagramme. Cette approche modulaire facilite la mise au point, la compréhension et la maintenance des applications.

Création d’un sous-VI

OBJECTIF Générer une icône et un connecteur pour le VI **My Thermometer** que vous avez construit au chapitre 1 et utiliser ce VI comme sous-VI.

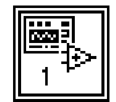
Pour utiliser un VI comme sous-VI, vous devez créer une icône pour le représenter dans le diagramme d’un autre VI, ainsi qu’un cadre connecteur auquel vous pourrez connecter les entrées et les sorties.

L'icône



Créez l'icône qui représentera le VI dans le diagramme composé d'autres VIs. Une icône peut être une représentation graphique du VI, ou bien encore une description textuelle du VI ou de ses broches.

1. Si vous avez fermé le VI **My Thermometer**, ouvrez-le de nouveau en sélectionnant **File»Open...**.
2. Choisissez `My Thermometer.vi` dans le répertoire `mywork.llb`.

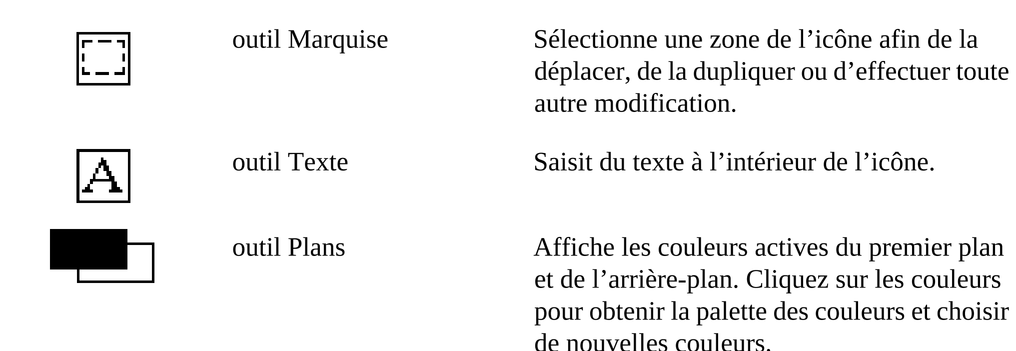


3. Appelez l'Editeur d'icônes en ouvrant un menu local sur le cadre icône situé en haut à droite de la face-avant et en choisissant **Edit Icon**. En guise de raccourci, vous pouvez également double-cliquer sur le cadre icône pour l'éditer.

Outils et boutons de l'Editeur d'icônes

Les outils situés à gauche de la zone d'édition exécutent les fonctions suivantes :

outil Crayon	Dessine et efface, pixel par pixel.
outil Trait	Trace des lignes droites. Appuyez sur la touche <Maj> puis faites glisser cet outil pour tracer des lignes horizontales, verticales ou obliques.
outil Pipette	Copie la couleur du premier plan d'un élément de l'icône.
outil Pot de peinture	Remplit une zone délimitée avec la couleur du premier plan.
outil Rectangle	Dessine un rectangle avec la couleur du premier plan. Double-cliquez sur cet outil pour tracer le cadre de l'icône avec la couleur du premier plan.
outil Rectangle plein	Dessine un rectangle avec la couleur du premier plan et le remplit de la couleur de l'arrière-plan. Double-cliquez pour tracer le cadre de l'icône avec la couleur du premier plan et colorier le fond avec la couleur de l'arrière-plan.



	outil Marquise	Sélectionne une zone de l’icône afin de la déplacer, de la dupliquer ou d’effectuer toute autre modification.
	outil Texte	Saisit du texte à l’intérieur de l’icône.
	outil Plans	Affiche les couleurs actives du premier plan et de l’arrière-plan. Cliquez sur les couleurs pour obtenir la palette des couleurs et choisir de nouvelles couleurs.

Les boutons situés à droite de l’écran d’édition exécutent les fonctions suivantes :

Undo Annule la dernière opération effectuée.

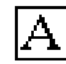
OK Enregistre votre dessin en tant qu’icône du VI et retourne à la fenêtre de la face-avant.

Cancel Retourne à la fenêtre de la face-avant sans tenir compte des modifications.

4. Supprimez l’icône par défaut.
   a. A l’aide de l’outil Marquise, choisissez la section intérieure de l’icône par défaut, représentée à gauche.
   b. Appuyez sur la touche `<Suppr>` pour supprimer l’intérieur de l’icône par défaut.

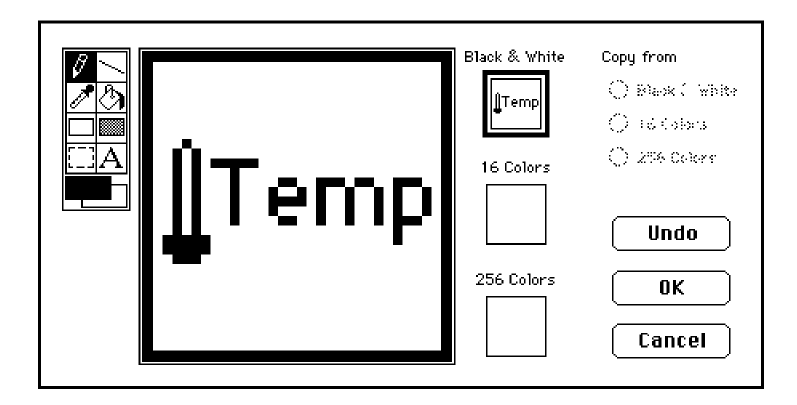


5. Dessinez le thermomètre avec l’outil Crayon.



6. Saisissez le texte avec l’outil Texte. Pour modifier la police, double-cliquez sur l’outil Texte. Exercez-vous à utiliser l’Editeur d’icônes.

Votre icône devrait ressembler à celle de l'illustration suivante.

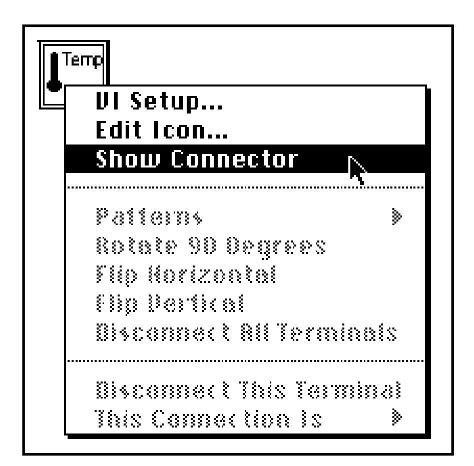


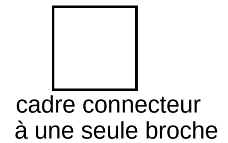
7. Fermez l'Editeur d'icônes en cliquant sur le bouton **OK** une fois l'icône terminée. La nouvelle icône apparaîtra dans le cadre icône en haut à droite de la face-avant.

Le connecteur

Maintenant, vous allez créer le connecteur.

1. Définissez la forme de la broche du connecteur en ouvrant un menu local sur le cadre icône de la face-avant et en choisissant **Show Connector**, conformément à l'illustration suivante.





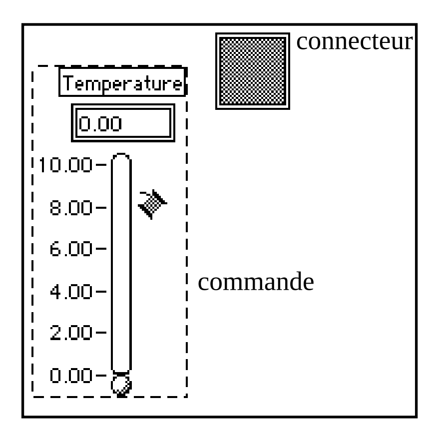
Comme LabVIEW sélectionne automatiquement un type de brochage en fonction du nombre de commandes et d'indicateurs disponibles sur la face-avant, il n'existe qu'une seule broche : l'indicateur thermomètre.

2. Affectez cette broche au thermomètre.



   a. Cliquez sur la broche du connecteur. Le curseur se transforme alors en outil Bobine et la broche s'assombrit.

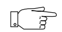
   b. Cliquez sur le thermomètre. Une ligne en tirets mobile entoure alors l'indicateur, comme dans l'illustration suivante.



Si vous cliquez sur une zone libre de la face-avant, la ligne en tirets disparaît et la broche sélectionnée passe au gris clair pour indiquer que vous avez affecté l'indicateur à cette broche. Une broche qui reste blanche indique une connexion défectueuse. Au besoin, répétez les étapes précédentes.

3. Enregistrez le VI en choisissant **File»Save**. Sur Macintosh, si vous utilisez la boîte de dialogue du fichier source pour enregistrer le VI dans une bibliothèque de VIs, vous devez cliquer sur le bouton **Use LLBs** avant de sélectionner la bibliothèque de VIs.

Ce VI est maintenant terminé et prêt à servir de sous-VI dans d'autres VIs. Son icône va pouvoir le représenter dans les diagrammes de VIs appelants. Son connecteur (mono-broche) donnera en sortie la température mesurée.



Remarque : ***le connecteur indique les entrées et les sorties d'un VI lorsque ce dernier est utilisé comme sous-VI. N'oubliez pas que les commandes de la face-avant ne peuvent s'utiliser qu'en entrée, alors que les indicateurs de la face-avant ne peuvent servir qu'en sortie.***

4. Fermez le VI en choisissant **File»Close.**

Mise en œuvre d'un VI en tant que sous-VI

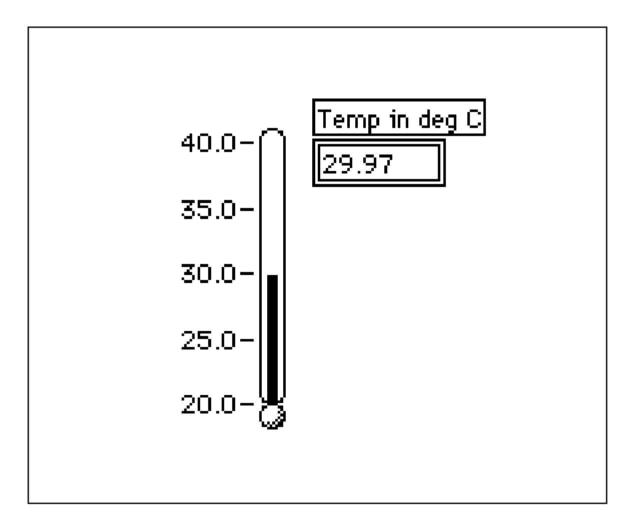
Vous pouvez utiliser n'importe quel VI doté d'une icône et d'un connecteur comme sous-VI dans le diagramme d'un autre VI. Pour choisir les VIs à utiliser comme sous-VIs, il suffit de sélectionner **Functions»Select a VI...**. Le fait de choisir cette option ouvre une boîte de dialogue fichier, à partir de laquelle vous pouvez sélectionner n'importe quel VI dans le système. Si vous appelez un VI qui n'a pas d'icône ni de connecteur, un carré blanc s'affiche alors dans le diagramme du VI appelant, que vous ne pouvez pas câbler.

Un sous-VI équivaut à un sous-programme et un nœud de sous-VI (icône/connecteur) à l'appel d'un sous-programme. Le nœud du sous-VI ne constitue pas le sous-VI de même qu'une instruction d'appel de sous-programme dans un programme ne constitue pas le sous-programme lui-même. Un diagramme qui contient plusieurs nœuds identiques appelle plusieurs fois le sous-VI correspondant.

OBJECTIF Construire un VI qui utilise le VI **My Thermometer** comme sous-VI.

Le VI **My Thermometer** que vous avez construit donne une température exprimée en degrés Fahrenheit. Vous utiliserez ce relevé et convertirez la température en degrés Celcius.

La face-avant



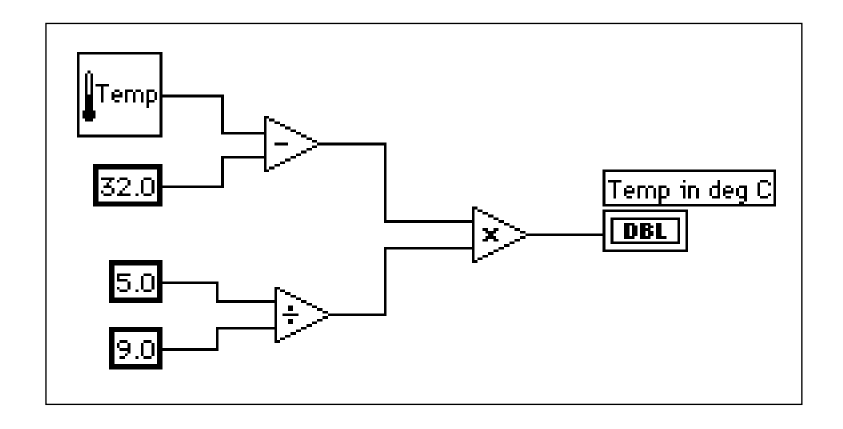
1. Ouvrez une nouvelle face-avant en choisissant **File»New**.
2. Choisissez le thermomètre dans **Controls»Numeric**. Donnez-lui l'étiquette `Temp in deg C`.
3. Modifiez l'échelle du thermomètre pour l'adapter aux valeurs à mesurer. A l'aide de l'outil Doigt, double-cliquez sur la valeur minimale, tapez `20`, puis appuyez sur le bouton `<Enter>` du clavier numérique. Il n'est pas nécessaire de saisir les décimales et les zéros après la virgule. LabVIEW les ajoute automatiquement lorsque vous validez votre saisie. De la même façon, fixez la valeur maximale à 40 et appuyez sur le bouton `<Enter>` du clavier numérique. LabVIEW ajuste automatiquement les valeurs intermédiaires.



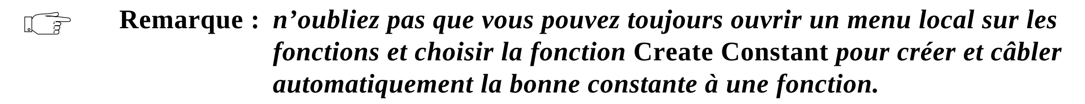
A chaque fois que vous créez une nouvelle commande ou un nouvel indicateur, LabVIEW crée le terminal correspondant dans le diagramme. Les symboles des terminaux rappellent le type de données de la commande ou de l'indicateur. Par exemple, un terminal DBL représente un nombre à virgule flottante en double précision.

Le diagramme

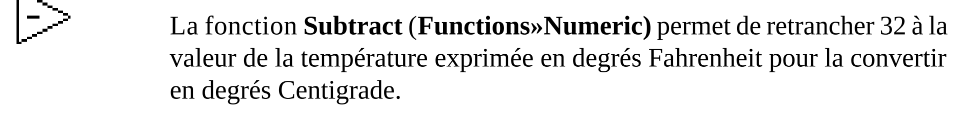
1. Choisissez **Windows»Show Diagram.**
2. Ouvrez un menu local dans une zone libre du diagramme et choisissez **Functions»Select a VI...**. Une boîte de dialogue s'affiche. Repérez puis ouvrez la bibliothèque `mywork.llb`. Double-cliquez sur `My Thermometer.vi` ou mettez-le en surbrillance, puis cliquez sur **Open** dans la boîte de dialogue. LabVIEW positionne le VI **My Thermometer** dans le diagramme.
3. Ajoutez les autres éléments du diagramme conformément à l'illustration suivante.



Numeric constant (**Functions»Numeric)**. Ajoutez trois constantes numériques dans le diagramme en leur affectant les valeurs 32.0, 5.0, et 9.0 avec l'outil Texte.



Remarque : ***n'oubliez pas que vous pouvez toujours ouvrir un menu local sur les fonctions et choisir la fonction* Create Constant *pour créer et câbler automatiquement la bonne constante à une fonction.***



La fonction **Subtract** (**Functions»Numeric)** permet de retrancher 32 à la valeur de la température exprimée en degrés Fahrenheit pour la convertir en degrés Centigrade.

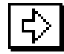
La fonction **Divide** (**Functions»Numeric)** calcule la valeur de 5/9 pour permettre la conversion de température.

La fonction **Multiply** (**Functions»Numeric)** délivre la valeur de la température en degrés Centigrade à partir du processus de conversion.

4. Câblez les éléments du diagramme conformément à l'illustration précédente.



Remarque : ***un fil brisé entre l'icône Thermometer et le terminal Temp in deg C peut indiquer que vous avez mal affecté le connecteur du sous-VI à l'indicateur de sa face-avant. Pour y remédier, veuillez vous reporter aux instructions données dans la section*** *Création d'un sous-VI* ***au début de ce chapitre. Lorsque vous aurez modifié le sous-VI, vous aurez peut-être à sélectionner*** **Relink to SubVI** ***dans le menu local de l'icône. Si nécessaire, choisissez*** **Edit»Remove Bad Wires.**



5. Revenez dans la face-avant puis cliquez sur le bouton Exécution dans la barre d'outils.

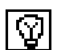
La barre d'outils du diagramme

Le diagramme contient des options supplémentaires qui ne figurent pas dans la barre d'outils de la face-avant.

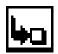
Barre d'outils du diagramme :



La barre d'outils du diagramme contient les boutons suivants que vous pouvez utiliser pour mettre au point les VIs.



bouton Ampoule : affiche les données passant dans les fils de liaison



bouton Exécution détaillée : exécution pas à pas détaillée (dans les boucles, sous-VIs, etc.)



bouton Exécution semi-détaillée : exécution pas à pas, chaque boucle, sous-VI, etc., étant considéré comme un seul pas



bouton Sortie : termine l'exécution en cours de la boucle, du VI, du diagramme, etc.

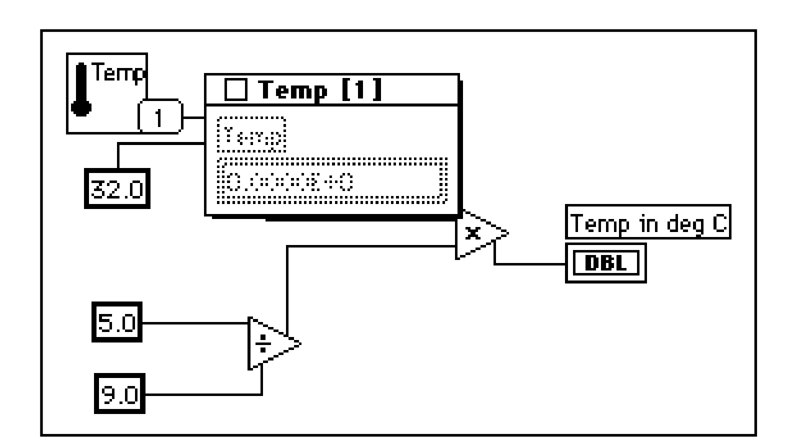
Quelques techniques de mise au point

Le rôle du thermomètre est d'afficher une valeur dans la gamme de mesure choisie. Cependant, si vous souhaitez par exemple, obtenir la valeur en Fahrenheit pour pouvoir faire une comparaison et une mise au point, LabVIEW offre un certain nombre d'outils qui peuvent vous y aider. Dans cet exercice, vous allez étudier la sonde et le mode Animation. Ces deux techniques ainsi que d'autres outils et astuces de mise au point sont repris en détail au chapitre 9, *Les techniques et astuces de programmation* et de mise au point, de ce tutorial.

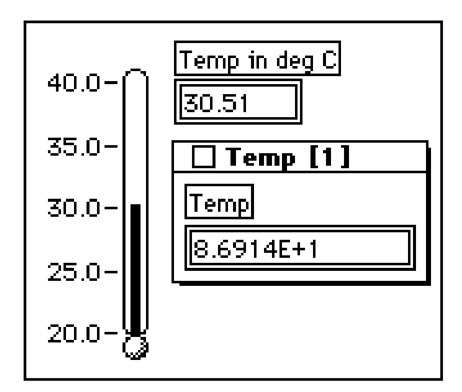
1. Choisissez **Windows»Show Diagram**.



2. Choisissez l'outil Sonde dans la palette **Tools**. A l'aide de l'outil Sonde, cliquez sur la valeur (fil) température du sous-VI **My Thermometer**. Une fenêtre avec une sonde s'affiche alors à l'écran avec le titre `Temp 1` et un glyphe jaune avec le numéro de la sonde, conformément à la représentation suivante. La fenêtre Sonde apparaît également dans la face-avant.



3. Retournez dans la face-avant. Déplacez la fenêtre de la sonde de manière à visualiser les valeurs de la sonde et celles du thermomètre conformément à l'illustration suivante. Lancez le VI. La température exprimée en degrés Fahrenheit s'affiche dans la fenêtre de la sonde.



4. Fermez la fenêtre de la sonde en cliquant sur le bouton de fermeture en haut à gauche de sa barre d'outils.

Une autre technique de mise au point très utile consiste à examiner le flux des données en utilisant le mode Animation de LabVIEW.

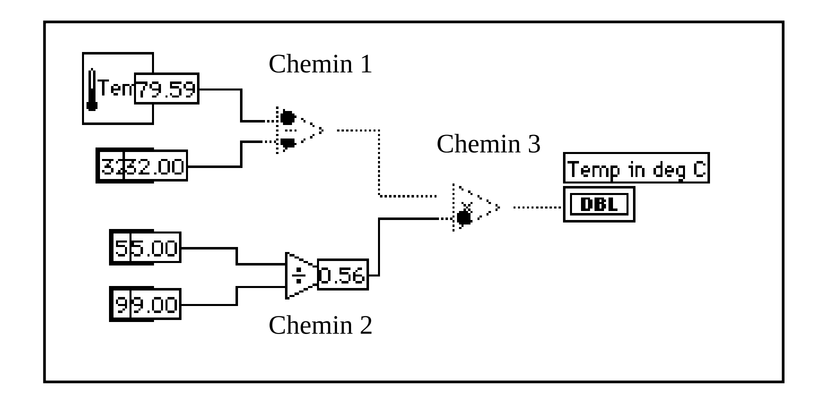
5. Revenez au diagramme du VI en choisissant **Windows»Show Diagram**.





6. Lancez le mode Animation en cliquant sur le bouton Ampoule de la barre d'outils, représenté en haut à gauche. Le bouton Ampoule se transforme alors en ampoule allumée.

7. Lancez le VI. Vous remarquerez que le diagramme s'anime. Les bulles mobiles représentent le flux des données qui traversent le VI. Vous remarquerez par ailleurs que les valeurs des données transmises apparaissent sur les fils de liaison et affichent en même temps les valeurs contenues dans ces fils, comme si vous aviez sondé le fil de liaison. C'est ce qu'illustre la figure suivante.

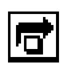


Remarquez l'ordre dans lequel s'exécutent les différents nœuds de LabVIEW. Dans un programme écrit en langage textuel classique, les instructions sont exécutées suivant leur ordre d'apparition dans le script du programme. Le logiciel LabVIEW lui, utilise la programmation par *flux des données*. Dans ce type de programmation, un nœud s'exécute uniquement lorsque toutes les données en entrée sont disponibles, et non pas en fonction de la disposition des entrées, de haut en bas ou de gauche à droite.

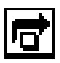
L'illustration précédente montre comment LabVIEW peut faire du multi-tâche entre les chemins 1 et 2 puisqu'il n'existe aucune dépendance entre les données. C'est-à-dire que rien dans le chemin 1 ne dépend des données qui circulent dans le chemin 2, et réciproquement. Le chemin 3 est le dernier à s'exécuter, car la fonction **Multiply** dépend des données obtenues à partir des fonctions **Subtract** et **Divide**.

Le mode Animation est un outil très utile pour étudier la nature des flux des données. Ce mode est repris en détail plus loin dans le chapitre 9, *Les techniques et astuces de programmation et de mise au point,* de ce tutorial.

Vous pouvez également utiliser les boutons d'exécution en mode pas à pas si vous souhaitez mieux contrôler le processus de mise au point.



8. Lancez l'exécution en mode pas à pas en cliquant sur le bouton Exécution semi-détaillée de la barre d'outils. Le seul fait de cliquer sur ce bouton affiche la première séquence qui sera exécutée dans le VI. Une fois que LabVIEW a terminé cette partie de la séquence, il met en évidence la prochaine tâche qui s'exécutera dans le VI.

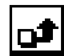


9. Passez à la fonction **Divide** en cliquant sur le bouton Exécution semi-détaillée de la barre d'outils. Le seul fait de cliquer sur ce bouton lance la fonction **Divide**. Une fois que LabVIEW a achevé cette tâche, il met en évidence la prochaine à s'exécuter dans le VI.





10. Allez dans le sous-VI **My Thermometer** en cliquant sur le bouton Exécution détaillée de la barre d'outils. Le seul fait de cliquer sur ce bouton ouvre la face-avant et le diagramme de votre sous-VI thermomètre. Vous avez maintenant le choix entre passer en mode Exécution pas à pas ou bien lancer le sous-VI.



11. Terminez l'exécution du diagramme en cliquant sur le bouton Sortie de la barre d'outils. Le seul fait de cliquer sur ce bouton termine toutes les séquences restant à mettre en œuvre dans le diagramme. Une fois que LabVIEW a terminé cette partie de la séquence, il met en évidence la prochaine tâche devant s'exécuter dans le VI. Vous pouvez également maintenir enfoncé le bouton de la souris lorsque vous cliquez sur le bouton Sortie pour accéder à un menu local. Dans ce menu local, vous pouvez limiter l'exécution du VI. L'illustration suivante affiche les options de fin d'exécution dans le menu local du bouton Sortie.



12. Enregistrez le VI dans le répertoire `mywork.llb`. Nommez-le `Using My Thermometer.vi`, puis fermez-le.

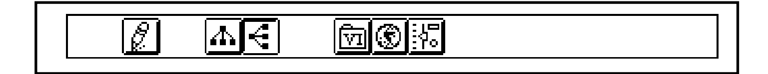
Ouverture, exécution et modification des sous-VIs

Vous pouvez ouvrir un VI comme sous-VI à partir du diagramme du VI appelant. Pour ce faire, double-cliquez sur l'icône du sous-VI ou choisissez **Project»This VI's SubVIs**. Ouvrez ensuite le diagramme en sélectionnant **Windows»Show Diagram**.

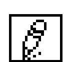
Sachez que toute modification effectuée sur un sous-VI n'affecte que la version en cours d'utilisation (en mémoire vive) jusqu'à ce que vous enregistriez le sous-VI. Vous remarquerez cependant que les modifications sont prises en compte pour tous les appels du sous-VI et qu'elles ne se limitent pas au nœud sur lequel vous étiez pour ouvrir le VI.

La fenêtre Hiérarchie

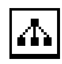
La fenêtre Hiérarchie (**Project»Show VI Hierarchy**) sert à visualiser les liens de dépendance des VIs en donnant des informations sur les VIs appelants et les sous-VIs. Cette fenêtre contient une barre d'outils qui va vous servir à configurer plusieurs types de paramètres pour les objets affichés. La figure ci-après présente la barre d'outils Hiérarchie du VI.



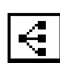
Vous pouvez utiliser les boutons de la barre d'outils de la fenêtre Hiérarchie ou le menu **VIEW**, ou bien encore ouvrir un menu local sur un espace vide de la fenêtre pour accéder aux options suivantes :



- Dessin : répartit les nœuds après des manipulations successives sur les nœuds de hiérarchies, pour réduire les croisements de lignes et privilégier l'aspect symétrique. S'il existe un nœud principal, vous pouvez alors parcourir la fenêtre de manière à ce que la première racine affichant des sous-VIs soit visible.



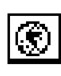
- Présentation de gauche à droite : présente les nœuds de haut en bas, les racines en haut.



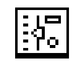
- Présentation de haut en bas : présente les nœuds de gauche à droite, les racines sur la gauche.



- Inclusion/Exclusion des VIs dans les bibliothèques de VIs : permute l'affichage du graphe déroulant pour masquer/afficher les VIs des bibliothèques de VIs.



- Inclusion/Exclusion des variables globales : permute l'affichage du graphe déroulant pour masquer/afficher les variables globales.



- Inclusion/Exclusion des types : permute l'affichage du graphe déroulant pour masquer/afficher les définitions de type.

De plus, le menu **View** et les menus locaux proposent les options **Show all VIs** et **Full VI Path in Label** auxquelles vous n'avez pas accès depuis la barre d'outils.

Au fur et à mesure que vous déplacez l'outil Doigt sur les objets dans la fenêtre Hiérarchie, LabVIEW affiche le nom du VI sous son icône.



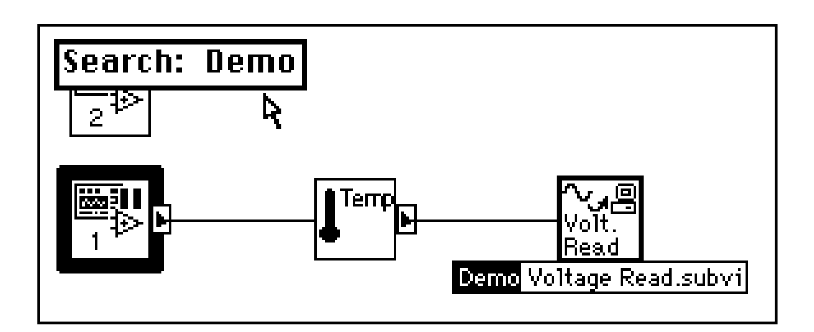
Utilisez la touche de commutation <Tab> située entre les outils Flèche et Fenêtre de défilement. Cette manipulation vous permettra de déplacer plus facilement les nœuds de la fenêtre Hiérarchie vers le diagramme.

Vous pouvez faire glisser le nœud d'un VI ou sous-VI dans un diagramme ou le copier dans le presse-papiers en cliquant sur le nœud. Tout en appuyant sur la touche <Maj>, cliquez sur le nœud d'un VI ou d'un sous-VI pour sélectionner plusieurs choix à recopier dans d'autres diagrammes ou faces-avant. Le fait de double-cliquer sur un nœud de VI ou de sous-VI suffit à ouvrir sa face-avant.

A tous les VIs ayant des sous-VIs correspond un bouton en forme de flèche qui permet de les afficher ou de les masquer. Le fait de cliquer sur la flèche rouge ou de double-cliquer sur le VI lui-même suffit à ouvrir les sous-VIs du VI. Une flèche noire sur un nœud d'un VI indique que tous les sous-VIs existants sont affichés. Vous pouvez également ouvrir un menu local sur le nœud d'un VI ou d'un sous-VI pour accéder à un menu déroulant avec des options comme celles permettant d'afficher ou de masquer les sous-VIs, d'ouvrir la face-avant d'un VI ou d'un sous-VI, d'éditer l'icône d'un VI, etc.

Recherche dans la hiérarchie

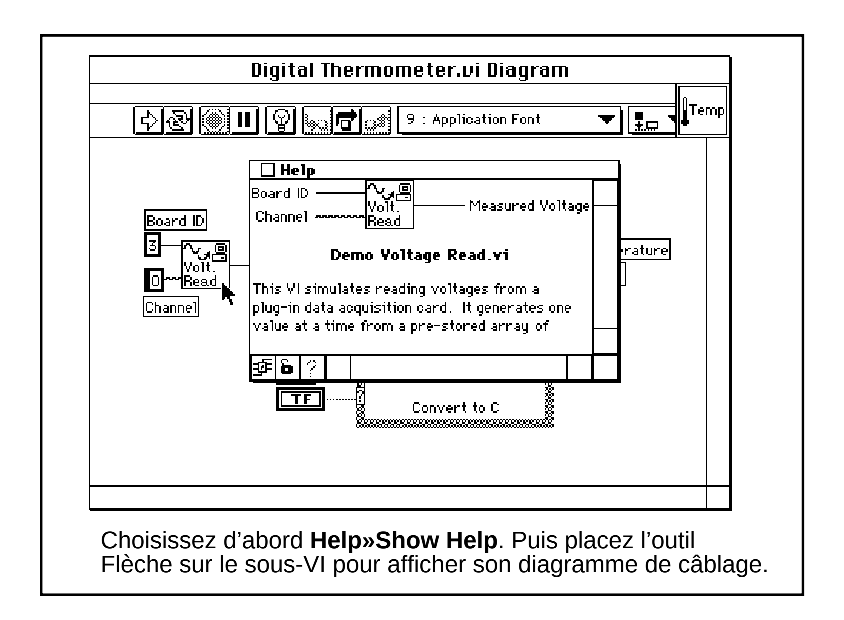
Vous pouvez aussi rechercher les nœuds visibles dans la fenêtre Hiérarchie par nom. Vous démarrez la recherche en tapant le nom du nœud n'importe où dans la fenêtre. Lorsque vous saisissez le texte, une fenêtre de recherche apparaît qui affiche le texte au fur et à mesure que vous le tapez et qui effectue parallèlement une recherche au sein de la hiérarchie. La figure suivante en est une illustration.



Lorsque vous avez trouvé le nœud qui convient, appuyez sur le bouton <Enter> pour rechercher le nœud suivant correspondant aux critères de recherche, ou appuyez sur les touches <Maj-Enter> (Windows) ; <Maj-return> (Macintosh) ; <Maj-Return> (Sun) ; ou <Maj-Enter> (HP-UX) pour trouver le nœud précédent correspondant aux critères de recherche.

L'aide en ligne pour les nœuds des sous-VIs

Lorsque vous placez l'un des outils sur le nœud du sous-VI, la fenêtre d'aide affiche l'icône du sous-VI avec les fils rattachés à chaque broche. L'illustration suivante vous présente un exemple d'aide en ligne. Il s'agit du VI **Digital Thermometer** de la palette **Functions»Tutorial**. Votre VI **Thermometer** contient également le texte que vous avez saisi dans la boîte de dialogue relative aux informations du VI.



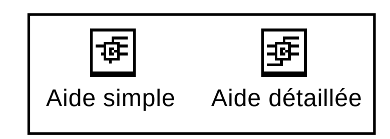
Choisissez d'abord **Help»Show Help**. Puis placez l'outil Flèche sur le sous-VI pour afficher son diagramme de câblage.

Aide simple/détaillée

Dans la fenêtre d'aide, vous avez la possibilité d'opter pour une représentation simple ou, à l'inverse, détaillée des objets du diagramme.

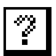
☞ **Remarque :** ***lorsque vous ouvrez la fenêtre d'aide, LabVIEW affiche par défaut la représentation simple.***

Dans le cas de la représentation simple, LabVIEW n'affiche que les entrées obligatoires et recommandées pour les VIs et les fonctions. Dans la représentation détaillée, LabVIEW affiche aussi les entrées dites optionnelles, plus le nom complet du chemin du VI. Pour accéder à une représentation simple, appuyez sur le bouton Aide simple/ détaillée ou choisissez **Help»Simple Diagram Help**. La figure suivante est l'illustration des deux interrupteurs à utiliser pour obtenir une représentation simple ou détaillée.

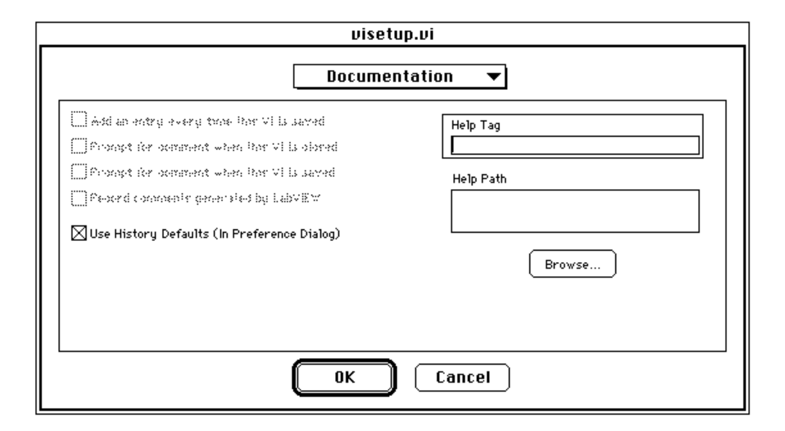


Dans la fenêtre d'aide, les entrées obligatoires figurent en gras, les entrées recommandées en texte normal et les données optionnelles en gris. Lorsque vous développez vos propres VIs, vous avez la possibilité de déterminer quelles entrées seront obligatoires, recommandées ou optionnelles en ouvrant un menu local sur une entrée ou sortie du cadre connecteur et en choisissant l'option qui convient dans le sous-menu **This Connection is**.

Liens vers les fichiers d'aide en ligne



Dans la fenêtre d'aide, vous pouvez cliquer sur le bouton d'aide en ligne pour accéder à l'aide en ligne de LabVIEW ainsi qu'aux fichiers d'aide que vous avez créés avec le compilateur d'aide. Si vous souhaitez créer votre propre fichier d'aide, vous devez en indiquer le lien en cliquant sur le cadre icône et en choisissant **VI Setup...**. Lorsque la boîte de dialogue **VI Setup** s'ouvre, choisissez **Documentation** dans la commande de type roue codeuse située en haut de la boîte, puis entrez le chemin correspondant au fichier d'aide dans la zone **Help Path**. L'exemple suivant présente les options disponibles dans la boîte de dialogue **VI Setup**.



Vous choisissez **Browse...** pour associer le fichier d'aide et l'objet à votre VI.

Pour plus d'informations sur la création des fichiers d'aide, veuillez vous reporter à la section *La création de vos propres fichiers d'aide* du chapitre 25, *La gestion de vos applications*, du *Manuel de l'utilisateur LabVIEW*.

Résumé

Le fait de pouvoir appeler des VIs à l'intérieur de VIs de niveau supérieur permet de construire des diagrammes modulaires. Cette modularité facilite la lecture des diagrammes et simplifie leur mise au point.

Un VI servant de sous-VI doit obligatoirement posséder une icône et un connecteur. Les broches de son connecteur permettent de véhiculer des données d'entrée à son code exécutable et de récupérer en sortie les résultats après exécution du code.

Vous créez l'icône à l'aide de l'Editeur d'icônes. Vous définissez le connecteur en choisissant le nombre de broches nécessaires au VI, puis en affectant à chacune une commande ou un indicateur de face-avant.

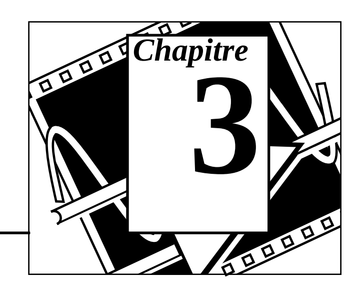
Une fois l'icône et le connecteur créés, le VI peut servir de sous-VI à un autre VI. Vous choisissez les sous-VIs à l'aide de la palette **Functions»Select a VI...**.

LabVIEW propose plusieurs outils pour mettre au point les VIs. Vous pouvez placer des sondes sur n'importe quel fil et afficher les valeurs qui les traversent lorsque le VI fonctionne. Le mode Animation anime le diagramme en affichant le flux des données sous la forme de bulles mobiles et d'auto-sondes. Vous avez la possibilité de recourir au mode pas à pas pour mettre au point les VIs et étudier le flux des données qui circulent dans les VIs et sous-VIs. Ces techniques de mise au point et d'autres encore sont décrites plus loin dans le chapitre 9, *Les techniques et astuces de programmation et de mise au point*, de ce tutorial.

Vous utilisez la fenêtre Hiérarchie pour visualiser sous forme graphique les liens de dépendance des VIs et des sous-VIs. Avec cette fenêtre, vous pouvez choisir entre plusieurs représentations du VI, demander une représentation avec des informations sur les définitions des types, les variables globales, etc. Pour accéder à la fenêtre Hiérarchie, choisissez **Project»Show VI Hierarchy**.

LabVIEW fournit par ailleurs une aide en ligne pour les sous-VIs. Cette aide en ligne vous aidera à câbler correctement les sous-VIs. Vous pouvez également utiliser l'aide en ligne pour obtenir une représentation simple ou à l'inverse détaillée d'un VI ou d'un sous-VI.

Boucles et graphes déroulants



Vous allez apprendre :

- Comment utiliser une boucle *While*.
- Comment afficher des données dans un graphe déroulant.
- Ce qu'est un registre à décalage et comment l'utiliser.
- Comment utiliser une boucle *For*.

Les structures contrôlent le flux des données dans les VIs. LabVIEW s'articule autour de quatre structures différentes : la boucle *While*, la boucle *For*, la structure Condition et la structure Séquence. Ce chapitre aborde les boucles *While* et *For* ainsi que les graphes déroulants et les registres à décalage. Les structures Condition et Séquence sont détaillées au chapitre 5, *Structures Condition, structures Séquence et boîte de calcul*.

Pour consulter des exemples de structures, veuillez vous reporter au répertoire `examples\general\structs.llb`.

Pour consulter des exemples de graphes déroulants, veuillez vous reporter au répertoire `examples\general\graphs\charts.llb`.

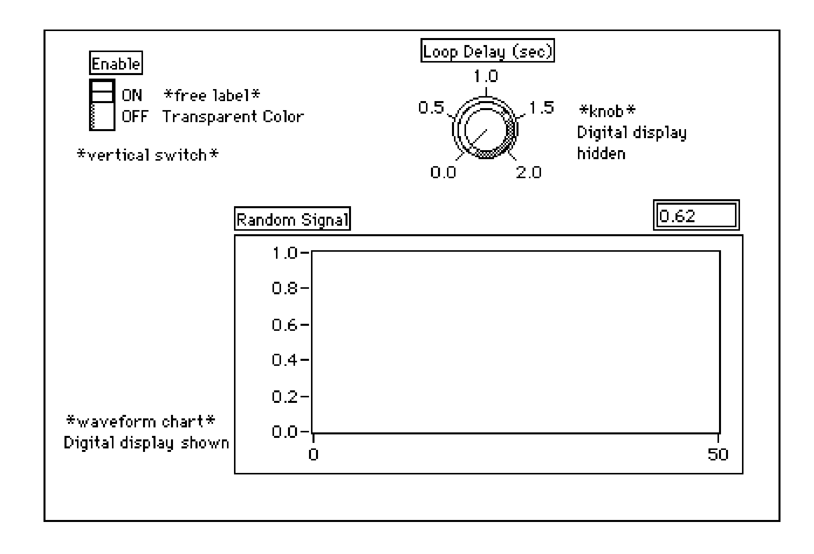
Mise en œuvre d'une boucle *While* et d'un graphe déroulant

OBJECTIF Utiliser une boucle *While* et un graphe déroulant pour acquérir et présenter les données en temps réel.

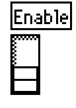
Vous allez construire un VI qui génère des données aléatoires et les restitue ensuite dans un graphe déroulant. Sur sa face-avant, le VI disposera d'un bouton rotatif pour régler le temps de scrutation de la boucle entre 0 et 2 secondes et d'un interrupteur pour arrêter le VI. Vous apprendrez à modifier le comportement mécanique de l'interrupteur pour ne plus avoir à l'actionner à chaque mise en route

du VI. Inspirez-vous de la face-avant de l'illustration suivante pour construire votre VI.

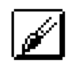
La face-avant



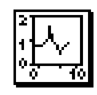
1. Ouvrez une nouvelle face-avant.



2. Placez un interrupteur vertical (**Controls»Boolean**) dans la face-avant. Nommez-le `Enable`. Il vous servira à interrompre le processus d'acquisition.

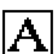


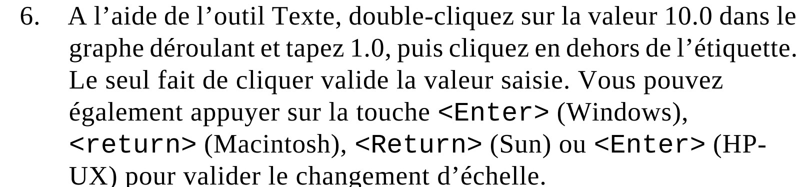
3. Utilisez l'outil Texte pour créer les textes libres MARCHE et ARRET. Utilisez l'outil Pinceau pour effacer les bords des textes libres. La lettre `T` en bas à gauche de la palette de couleurs permet de rendre un objet transparent.



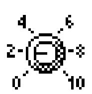
4. Placez un graphe déroulant (**Controls»Graph**) dans la face-avant. Nommez-le `Random Signal`. Ce graphe servira à afficher les données aléatoires en temps réel.

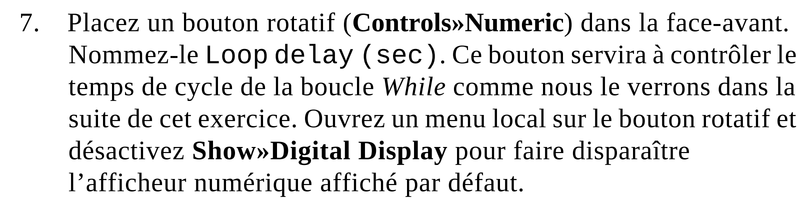
5. Ouvrez un menu local sur le graphe, puis choisissez **Show»Digital Display**. Cet afficheur numérique contient la dernière valeur.



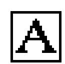



6. A l'aide de l'outil Texte, double-cliquez sur la valeur 10.0 dans le graphe déroulant et tapez 1.0, puis cliquez en dehors de l'étiquette. Le seul fait de cliquer valide la valeur saisie. Vous pouvez également appuyer sur la touche `<Enter>` (Windows), `<return>` (Macintosh), `<Return>` (Sun) ou `<Enter>` (HP-UX) pour valider le changement d'échelle.



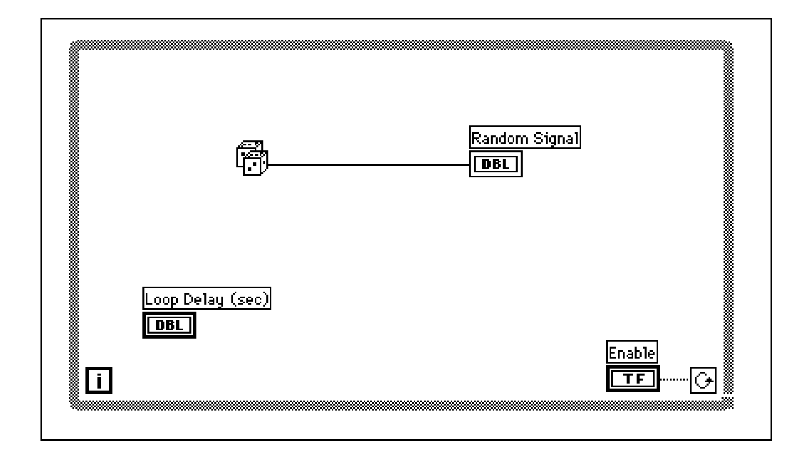


7. Placez un bouton rotatif (**Controls»Numeric**) dans la face-avant. Nommez-le `Loop delay (sec)`. Ce bouton servira à contrôler le temps de cycle de la boucle *While* comme nous le verrons dans la suite de cet exercice. Ouvrez un menu local sur le bouton rotatif et désactivez **Show»Digital Display** pour faire disparaître l'afficheur numérique affiché par défaut.



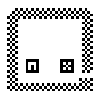
8. A l'aide de l'outil Texte, double-cliquez sur la valeur `10.0` dans l'échelle qui entoure le bouton rotatif, tapez `2.0`, puis cliquez en dehors de l'étiquette pour valider la nouvelle valeur.

Le diagramme





1. Ouvrez la fenêtre du diagramme.

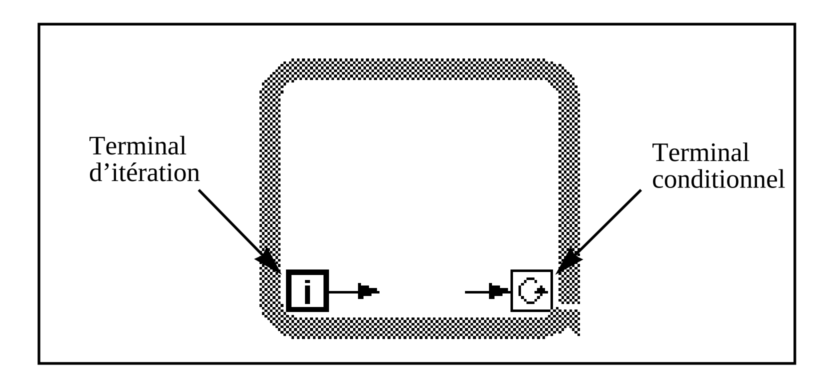


2. Placez-y la boucle *While*, en la choisissant dans la palette **Functions»Structures**. Cette boucle *While* est une boîte réajustable qui n'est pas immédiatement déposée dans le diagramme. Au contraire, vous avez ici la possibilité de la positionner et de la redimensionner à votre guise. Pour ce faire, cliquez dans une zone en haut à gauche de tous les terminaux. Continuez, tout en maintenant le bouton de la souris enfoncé, et dessinez un rectangle de manière à entourer tous ces terminaux. Une boucle *While* est alors créée à l'emplacement et au format voulus.

Terminal conditionnel

Terminal d'itération

La boucle *While*, reproduite dans l'exemple suivant, est une boîte réajustable que vous utilisez pour exécuter le diagramme qu'elle contient, jusqu'à ce que la valeur booléenne transmise au *terminal conditionnel* (un terminal d'entrée) prenne la valeur FALSE. Le VI vérifie la valeur fournie à la fin de chaque itération de la boucle *While*, si bien que la boucle *While s'exécute toujours au moins une fois*. Le *terminal d'itération* est une sortie numérique qui contient le nombre de fois que la boucle s'est exécutée. Cependant, le comptage des itérations démarre toujours à zéro, moyennant quoi, lorsque la boucle ne s'exécute qu'une seule fois, le terminal d'itération de sortie vaut `0`.

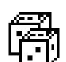


La boucle *While* équivaut au pseudo-code suivant :

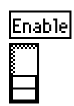
```
Faire

Exécuter le diagramme à l'intérieur de la boucle (ce qui
définit la condition)

Tant que la condition While est VRAIE
```



3. Choisissez la fonction **Random Number (0-1)** dans la palette **Functions»Numeric**.
4. Câblez le diagramme conformément à l'illustration de la section *Diagramme*, en connectant la fonction **Random Number (0-1)** au terminal du graphe déroulant **Random Signal**, et l'interrupteur Enable au terminal conditionnel de la boucle *While*. Ne câblez pas pour l'instant le terminal `Loop Delay`.



5. Retournez dans la face-avant et positionnez l'interrupteur vertical sur MARCHE en cliquant dessus avec l'outil Doigt. Lancez le VI.

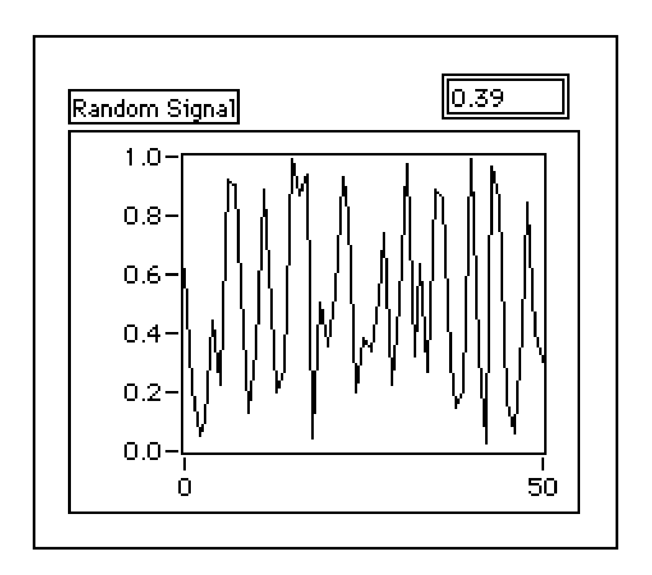
   La boucle *While* est une structure de bouclage indéfinie. Le diagramme qu'elle contient s'exécute tant que la condition indiquée reste vraie. Dans cet exemple, tant que l'interrupteur est sur la

position MARCHE (TRUE), le diagramme continue à générer des nombres aléatoires et à les afficher dans le graphe déroulant.

6. Pour arrêter la boucle, cliquez sur l'interrupteur vertical. Il passe alors en position ARRET et transmet la valeur FALSE au terminal conditionnel de la boucle qui s'arrête.
7. Le graphe déroulant dispose d'une mémoire tampon d'affichage dans laquelle sont conservés un certain nombre de points, une fois sortis de la zone d'affichage de l'écran. Affectez au graphe déroulant une barre de défilement en ouvrant un menu local sur celui-ci et en choisissant **Show»Scrollbar**. Vous pouvez utiliser l'outil Flèche pour ajuster la taille et la position de la barre de défilement.

   Pour vous déplacer dans le graphe déroulant, cliquez et maintenez enfoncé le bouton de la souris sur l'une des flèches de la barre de défilement.

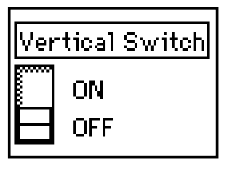
   Pour effacer le contenu de la mémoire tampon d'affichage et réinitialiser le graphe déroulant, ouvrez un menu local sur le graphe et choisissez **Data Operations»Clear Chart**.

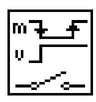


☞ **Remarque :** ***par défaut, la taille de la mémoire tampon d'affichage est de 1024 points. Vous pouvez augmenter ou diminuer la taille de la mémoire en ouvrant un menu local sur le graphe déroulant et en choisissant l'option*** **Chart History Length...** .

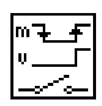
Comportement mécanique des interrupteurs booléens

Vous avez sans doute remarqué qu'à chaque fois que vous lancez le VI, il vous faut d'abord mettre l'interrupteur vertical sur la position MARCHE puis cliquer sur le bouton Exécution dans la barre d'outils. Avec LabVIEW, vous pouvez modifier le comportement mécanique des commandes booléennes. Vous disposez pour cela de six choix possibles pour l'action mécanique d'un élément de contrôle tout-ou-rien : **Switch When Pressed** (Commutation à l'appui), **Switch When Released** (Commutation au relâchement), **Switch Until Released** (Commutation jusqu'au relâchement), **Latch When Pressed** (Armement à l'appui), **Latch When Released** (Armement au relâchement), et **Latch Until Released** (Armement jusqu'au relâchement). LabVIEW fournit un exemple démontrant ces états ; il s'agit de `Mechanical Action of Booleans.vi` qui se trouve dans la bibliothèque `examples\general\controls\booleans.llb`. A titre d'exemple, prenez le cas de l'interrupteur vertical suivant. La valeur par défaut de cet interrupteur est ARRET (FALSE).

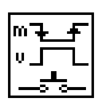




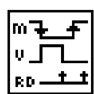
L'action **Switch When Pressed** (Commutation à l'appui) modifie la valeur de la commande à chaque fois que vous cliquez sur l'interrupteur avec l'outil Doigt. Son fonctionnement est comparable à celui d'un interrupteur de plafonnier. Peu importe le nombre de fois que le VI vient lire sa valeur.



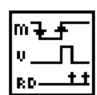
L'action **Switch When Released** (Commutation au relâchement) modifie la valeur de la commande seulement lorsque vous relâchez le bouton de la souris, après avoir cliqué à l'intérieur de la zone symbolisant l'interrupteur. Peu importe le nombre de fois que le VI vient lire sa valeur. Cette opération est comparable à ce qui se passe lorsque vous cliquez sur une case à cocher d'une boîte de dialogue. Celle-ci est alors mise en surbrillance mais ne change pas tant que vous n'avez pas relâché le bouton de la souris.



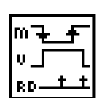
L’action **Switch Until Released** (Commutation jusqu’au relâchement) modifie la valeur de la commande chaque fois que vous cliquez sur cet interrupteur. Elle conserve sa nouvelle valeur jusqu’à ce que vous relâchiez le bouton de la souris, moment où la commande reprend sa valeur initiale. Cette action est comparable à celle d’une sonnette. Peu importe le nombre de fois que le VI vient lire l’état de la commande.



L’action **Latch When Pressed** (Armement à l’appui) modifie la valeur de la commande à chaque fois que vous cliquez dessus. Elle conserve sa nouvelle valeur jusqu’à ce que le VI vienne la lire. La commande retrouve alors sa valeur par défaut. (Cette action a lieu que vous continuiez ou non à appuyer sur le bouton de la souris) Cette action est comparable à celle d’un fusible et est très utile pour arrêter les boucles *While* ou pour demander au VI d’exécuter une opération une seule fois, chaque fois que vous cliquez sur l’interrupteur.



L’action **Latch When Released** (Armement au relâchement) modifie la valeur de la commande chaque fois que vous relâchez le bouton de la souris. Lorsque le VI lit la valeur, la commande reprend son ancienne valeur. Cette action permet de garantir une nouvelle valeur au moins. Comme pour l’action **Switch When Released**, cette action est semblable au comportement des boutons qui figurent dans les boîtes de dialogue. Le fait de cliquer dessus met en évidence le bouton et le fait de le relâcher fournit une valeur.



L’action **Latch Until Released** (Armement jusqu’au relâchement) modifie la valeur de la commande chaque fois que vous cliquez dessus. Elle conserve sa valeur jusqu’à ce que le VI vienne la lire ou jusqu’à ce que vous relâchiez le bouton de la souris, (c’est le dernier de ces deux événements qui est pris en compte).

1. Modifiez l’interrupteur vertical pour qu’il ne serve qu’à arrêter le VI. C’est-à-dire que vous n’aurez plus besoin de cliquer dessus pour exécuter le VI.
   a. Mettez-le sur la position MARCHE.
   b. Cliquez sur l’interrupteur pour ouvrir un menu local, puis choisissez **Data Operations»Make Current Value Default**. La position MARCHE devient alors la valeur par défaut.
   c. Cliquez sur l’interrupteur pour ouvrir un menu local, puis choisissez **Mechanical Action»Latch When Pressed**.

2. Lancez le VI. Cliquez sur l'interrupteur vertical pour interrompre le processus d'acquisition. L'interrupteur se met en position ARRET et reprend la position MARCHE lorsque le terminal conditionnel de la boucle *While* vient lire la valeur.

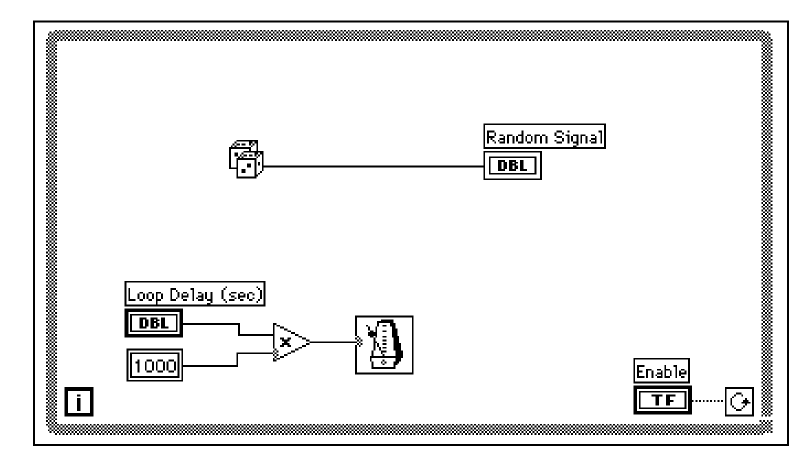
Temps de cycle

Lorsque vous demandez l'exécution du VI, la boucle *While* se déroule le plus rapidement possible. Cependant, vous avez peut-être besoin d'acquérir des données à intervalles réguliers, comme toutes les secondes ou toutes les minutes.

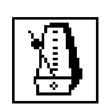
Le cas échéant, vous disposez des fonctions de temps de cycle de LabVIEW qui fournissent le temps en millisecondes (ms), ce qui ne signifie pas pour autant que votre système doit maintenir ce temps de cycle. Vous trouverez dans la liste suivante les instructions à suivre pour déterminer la précision des fonctions des temps de cycle de LabVIEW sur votre système.

- ***(Windows 3.1)*** L'horloge a une résolution par défaut de 55 ms. Vous pouvez configurer LabVIEW pour avoir une résolution de 1 ms en choisissant **Edit»Preferences...**, en choisissant **Performance** et **Disk** du menu **Chemins**, et en désactivant la case **Use Default Timer**. LabVIEW ne prend pas la résolution de 1 ms par défaut car cela implique une charge plus importante à votre système d'exploitation. Lisez la description relative à l'option **Use Default Timer** dans la section intitulée *Les préférences de performances et de disque* du chapitre 8, *La personnalisation de votre environnement*, du *Manuel de l'utilisateur LabVIEW* pour savoir si vous devez utiliser cette option.
- ***(Windows 95/NT)*** L'horloge a une résolution de 1 ms. Néanmoins, cela dépend du matériel utilisé. Par conséquent, dans des systèmes plus lents, comme un 80386, il y a de fortes chances que le temps de cycle de la résolution soit plus faible.
- ***(Macintosh)*** Pour les systèmes 68K qui ne disposent pas de l'extension QuickTime, l'horloge a une résolution de 16.66 ms (1/60ème de seconde). Si vous disposez d'un Power Macintosh ou si vous avez installé QuickTime dans votre système, la résolution de l'horloge est de 1 ms.
- ***(UNIX)*** L'horloge a une résolution de 1 ms.

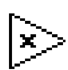
Vous pouvez contrôler le temps de cycle de la boucle en utilisant la fonction **Wait Until Next ms Multiple** (**Functions»Time & Dialog**). Cette fonction garantit qu’aucune itération ne durera moins longtemps que le nombre de millisecondes spécifié.



1. Modifiez le VI pour que la génération des nombres aléatoires se fasse à intervalles de temps réguliers. On en règle la durée au moyen du bouton rotatif, comme l’illustre le schéma précédent.



Fonction **Wait Until Next ms Multiple** (**Functions»Time & Dialog**). Dans cet exercice, cette fonction multiplie le terminal du bouton rotatif par 1000 pour convertir la valeur, exprimée en secondes, de ce bouton, en millisecondes. Reprenez cette valeur comme entrée à la fonction **Wait Until Next ms Multiple**.



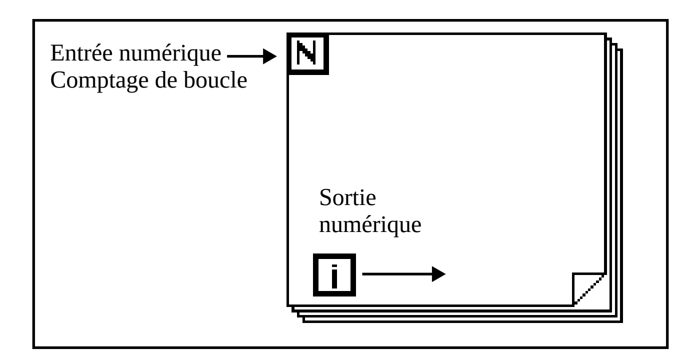
Fonction **Multiply** (**Functions»Numeric**). Dans cet exercice, la fonction **Multiply** permet de multiplier la valeur du bouton rotatif par 1000 pour convertir les secondes en millisecondes.



Numeric Constant(**Functions»Numeric**). La constante numérique contient la constante utilisée pour multiplier la valeur du bouton rotatif en vue d’obtenir une quantité exprimée en millisecondes. Ainsi donc, si le bouton affiche une valeur égale à 1.0, la boucle s’exécute toutes les 1000 millisecondes (c’est-à-dire une fois par seconde).

2. Exécutez le VI. Tournez le bouton afin d’obtenir des valeurs différentes pour le nombre de secondes.
3. Enregistrez et fermez le VI dans la bibliothèque `mywork.llb`. Nommez-le `My Random Signal.vi`.

La boucle *For*



Placez la boucle *For* dans le diagramme en la choisissant dans **Functions»Structures**. Une boucle *For* (voir l'illustration précédente) est une boîte réajustable, comme une boucle *While*. Comme la boucle *While*, elle n'est pas immédiatement déposée sur le diagramme. A la place, apparaît une petite icône qui représente la boucle *For* dans le diagramme. Vous pouvez la redimensionner et la déplacer à votre guise. Pour se faire, cliquez dans une zone à gauche de tous les terminaux. Tout en maintenant le bouton de la souris enfoncé, dessinez un rectangle qui englobe tous les terminaux que vous souhaitez faire apparaître à l'intérieur de la boucle *For*. Lorsque vous relâchez le bouton de la souris, LabVIEW crée une boucle *For* dans la taille et à l'emplacement voulus.

La boucle *For* exécute le diagramme dans les limites de sa bordure un certain nombre de fois préétabli. La boucle *For* compte deux terminaux :



le *terminal de comptage* (terminal d'entrée). Le terminal de comptage précise le nombre de fois que la boucle doit s'exécuter.

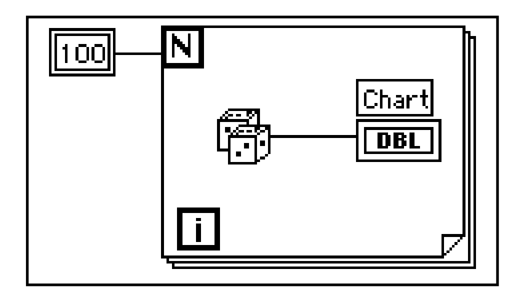


le *terminal d'itération* (terminal de sortie). Le terminal d'itération contient le nombre de fois que la boucle s'est déjà exécutée.

La boucle *For* équivaut au pseudo-code suivant :

```
Pour i = 0 jusqu'à N-1
Exécuter le diagramme à l'intérieur de la boucle
```

L'exemple suivant présente une boucle *For* qui génère 100 nombres aléatoires et affiche les points sur un graphe déroulant.



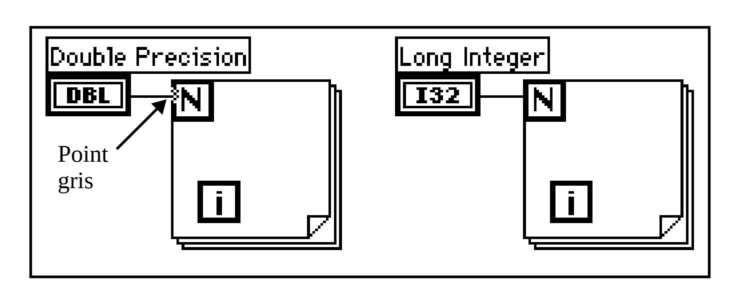
Conversion numérique

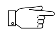
Jusqu'à présent, tous les indicateurs et commandes numériques que vous avez utilisés étaient des nombres à virgule flottante en double précision. Cependant, LabVIEW peut aussi traiter des nombres entiers (octets, mots ou entiers longs) ou des nombres à virgule flottante (simple précision, double précision ou précision étendue). Par défaut, les nombres sont traités comme des nombres à virgule flottante en double précision.

Si vous connectez deux terminaux de types différents, LabVIEW convertit l'un des terminaux dans le type de l'autre. A titre de rappel, LabVIEW place un point gris appelé point de coercition, sur le terminal objet de la conversion.



Par exemple, prenez le cas du terminal de comptage de la boucle *For*. La variable qui y figure est un entier long. Si vous connectez un nombre à virgule flottante en double précision à ce terminal, LabVIEW convertit ce nombre en entier long. Vous remarquerez qu'il y a un point gris sur le terminal de comptage de la boucle *For*.



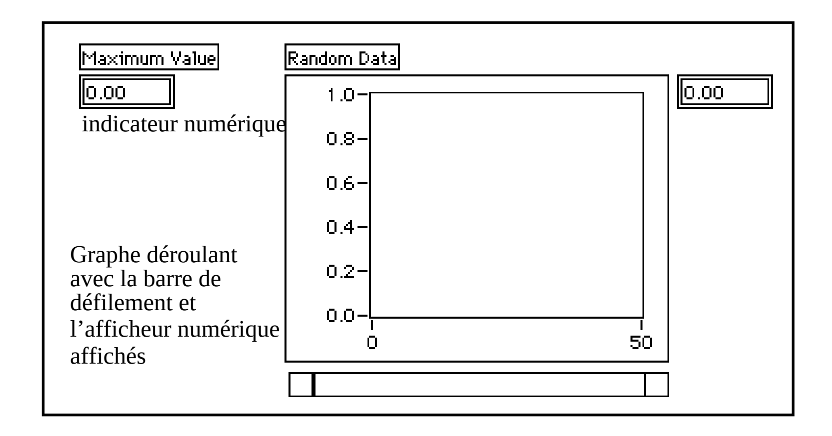


Remarque : ***lorsque le VI convertit un nombre à virgule flottante en un nombre entier, il l'arrondit à l'entier le plus proche. Si le nombre se situe entre deux entiers, il est arrondi à l'entier pair le plus proche. Par exemple, le VI arrondit 6.5 à 6, mais arrondit 7.5 à 8. Cette méthode correspond à la norme IEEE utilisée pour la lecture des nombres. Pour en savoir plus, veuillez vous reporter à la norme IEEE 754.***

Mise en œuvre d'une boucle *For*

OBJECTIF Utiliser une boucle *For* et des registres à décalage pour calculer la valeur maximale dans une série de nombres aléatoires. Vous allez mettre en œuvre une boucle *For* (N = 100) à la place de la boucle *While*.

La face-avant



1. Ouvrez une nouvelle face-avant et ajoutez-y les objets présentés dans l'illustration précédente.

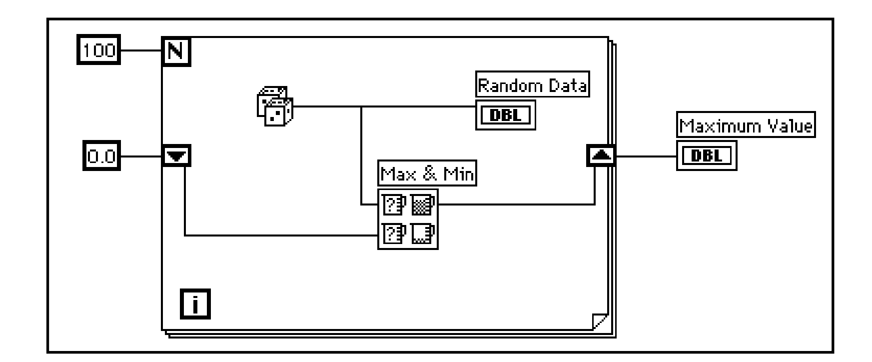


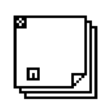
   a. Placez un indicateur numérique dans la face-avant et nommez-le `Maximum Value`.

   b. Placez un graphe déroulant dans la face-avant et nommez-le `Random Data`. Modifiez l'échelle du graphe déroulant afin d'avoir des valeurs allant de 0.0 à 1.0.

   c. Ouvrez un menu local dans le graphe déroulant et choisissez **Show»Scrollbar** puis **Show»Digital Display**.

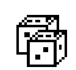
Le diagramme







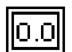
1. Ouvrez la fenêtre du diagramme.
2. Ajoutez la boucle *For* (**Functions»Structures**).
3. Ajoutez le registre à décalage en plaçant le curseur sur la bordure droite ou gauche de la boucle *For* et en choisissant **Add Shift Register**.
4. Ajoutez les autres éléments dans le diagramme.



Utilisez la fonction **Random Number (0-1)** (**Functions»Numeric**) pour générer les nombres aléatoires.

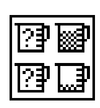


Numeric Constant (**Functions»Numeric**). La boucle *For* doit savoir combien d’itérations elle doit exécuter. Dans le cas présent, la boucle *For* s’exécutera 100 fois de suite.



Numeric Constant (**Functions»Numeric**). Sachant que la sortie du générateur de nombres aléatoires est comprise entre 0,0 et 1.0, vous allez régler le registre à décalage sur zéro.

Vous devez avoir un minimum de connaissances sur les données à acquérir pour initialiser un registre à décalage. Par exemple, il ne serait pas judicieux d’initialiser le registre à décalage sur 1.0, cette valeur étant déjà supérieure à toutes celles que vous allez acquérir. Si vous n’initialisez pas le registre à décalage, alors il devra contenir la valeur maximale de l’exécution précédente du VI. Par conséquent, vous pourriez très bien obtenir une valeur maximale en sortie qui n’ait aucun rapport avec les valeurs de l’acquisition en cours de collecte.



La fonction **Max & Min** (**Functions»Comparison**) prend deux valeurs numériques et restitue la plus grande en haut à droite et la plus petite en bas à droite. Comme dans cet exercice, seule la valeur la plus grande nous intéresse : ne connectez que la valeur la plus grande et ignorez la plus petite.

5. Câblez les terminaux conformément à la représentation suivante. Si le terminal Maximum Value se trouvait à l'intérieur de la boucle *For*, vous verriez qu'il serait continuellement mis à jour, mais comme il se trouve à l'extérieur, il ne contient que la dernière valeur maximale connue.



Remarque : ***la mise à jour des indicateurs à chaque itération de la boucle prend beaucoup de temps. C'est pourquoi il est préférable de l'éviter à chaque fois que possible pour améliorer la vitesse d'exécution.***

6. Lancez le VI.
7. Enregistrez le VI. Nommez-le `My Calculate Max.vi`.

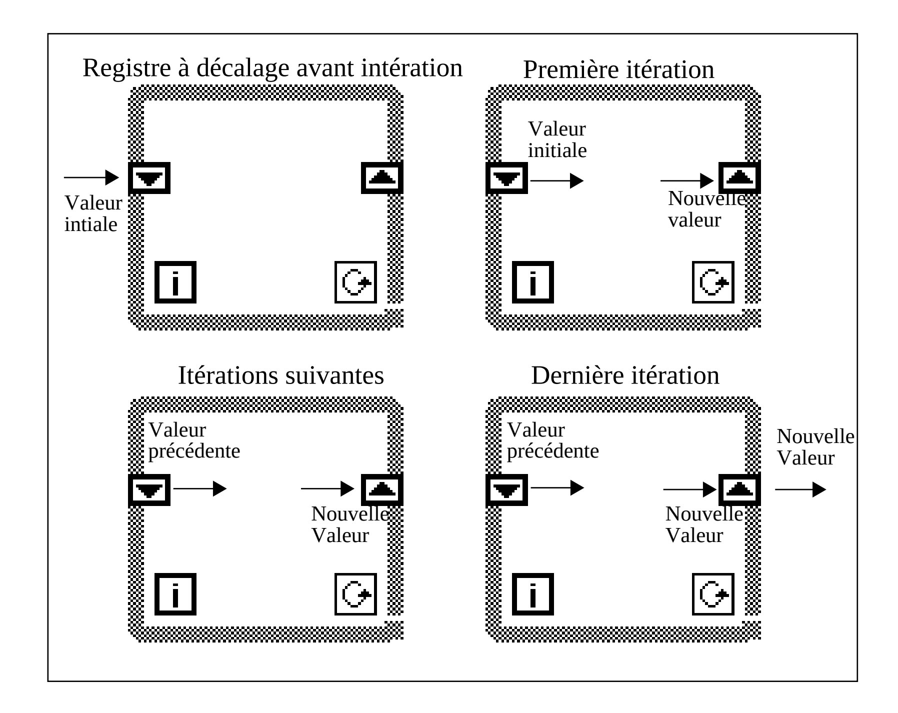
Les registres à décalage

Les registres à décalage (disponibles pour les boucles *While* et *For*) permettent de transférer les valeurs d'une itération de la boucle à une autre. Pour créer un registre à décalage, ouvrez un menu local sur la bordure droite ou gauche d'une boucle puis choisissez **Add Shift Register.**

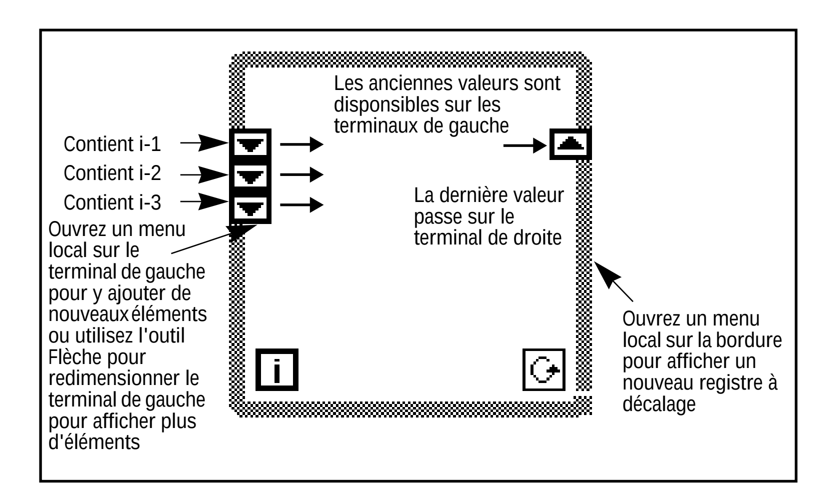


Le registre à décalage contient deux terminaux situés de part et d'autre des bords verticaux du cadre de la boucle. Le terminal *de droite* stocke les données pendant toute l'itération. A la fin de l'itération, elles se décalent et apparaissent dans le terminal *de gauche* au début de la

prochaine itération (voir l'illustration suivante). Un registre à décalage peut contenir n'importe quel type de données : numérique, booléen, chaîne de caractères, tableau, etc. Le registre à décalage s'adapte automatiquement au type de données du premier objet connecté au registre à décalage.



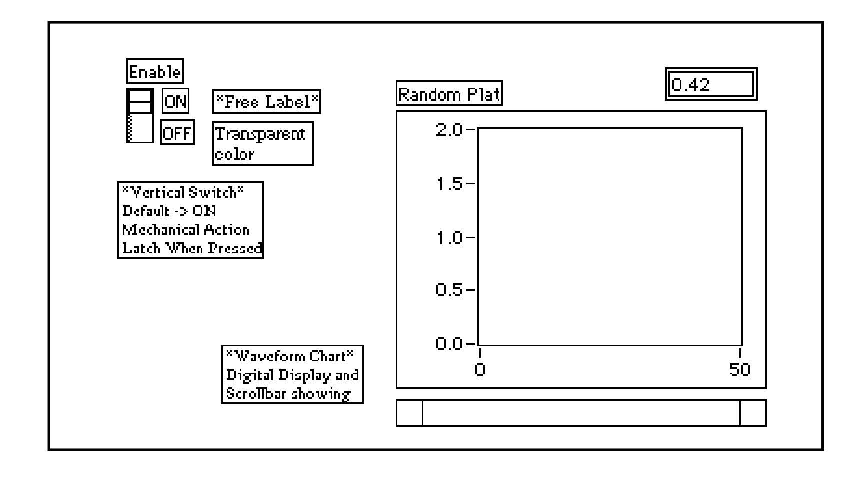
Vous pouvez configurer les registres à décalage de manière à ce qu'ils conservent les données d'itérations précédentes. Cette caractéristique est très utile pour établir la moyenne des points acquis lors d'itérations successives. Pour ce faire, il suffit de créer des terminaux supplémentaires en ouvrant un menu local sur le terminal de *gauche* ou de *droite* et en choisissant l'option **Add Element**. Par exemple, si un registre à décalage contient trois éléments dans son terminal de gauche, vous pouvez accéder à ces valeurs à partir des trois dernières itérations.



Mise en œuvre des registres à décalage

OBJECTIF Construire un VI qui affiche deux tracés aléatoires dans un graphe déroulant. Les deux tracés seront ceux d'un nombre aléatoire et de la moyenne des quatre derniers points du tracé aléatoire.

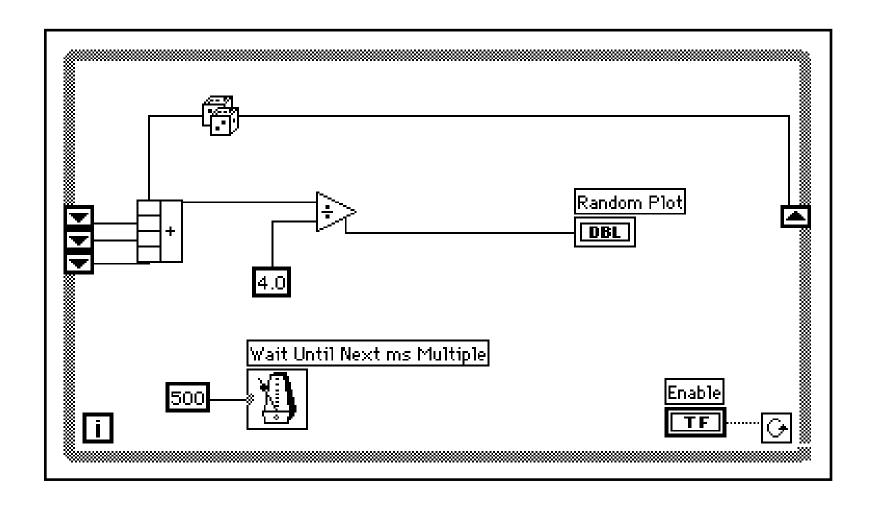
La face-avant



1. Ouvrez une nouvelle fenêtre pour y créer la face-avant reproduite dans l'illustration précédente.

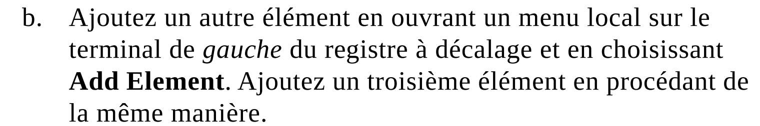
2. Après avoir ajouté le graphe déroulant dans la face-avant, modifiez l'échelle pour que les valeurs soient comprises entre 0.0 et 2.0.
3. Après avoir ajouté l'interrupteur vertical, définissez l'état MARCHE comme étant l'état par défaut puis configurez l'action mécanique en **Latch When Pressed**.

Le diagramme



1. Ajoutez **While Loop** (**Functions»Structures**) dans le diagramme et créez le registre à décalage.
   a. Ouvrez un menu local sur le bord droit ou gauche de la boucle *While* puis choisissez **Add Shift Register**.

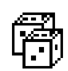




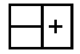
   b. Ajoutez un autre élément en ouvrant un menu local sur le terminal de *gauche* du registre à décalage et en choisissant **Add Element**. Ajoutez un troisième élément en procédant de la même manière.



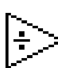
2. Construisez le diagramme de l'illustration précédente.



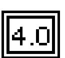
La fonction **Random Number (0-1)** (**Functions»Numeric**) génère des données brutes.



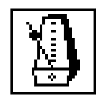
Fonction **Compound Arithmetic** (**Functions»Numeric**). Dans cet exercice, la fonction **Compound Arithmetic** renvoie la somme des nombres aléatoires de deux itérations. Pour ajouter plus de données en entrée, ouvrez un menu local et choisissez **Add Input** dans celui-ci.



Fonction **Divide** (**Functions»Numeric**). Dans cet exercice, la fonction **Divide** calcule la moyenne des quatre derniers nombres aléatoires.



Numeric Constant (**Functions»Numeric**). Lors de chaque itération de la boucle *While*, la fonction **Random Number (0-1)** génère une valeur aléatoire. Le VI ajoute cette valeur aux trois dernières valeurs conservées par les terminaux de gauche du registre à décalage. La fonction **Random Number (0-1)** divise le résultat par quatre pour calculer la moyenne des valeurs (la valeur actuelle et les trois précédentes). La moyenne est ensuite affichée dans le graphe déroulant.



Fonction **Wait Until Next ms Multiple** (**Functions»Time & Dialog**). Cette fonction permet de garantir que la durée de chaque itération de la boucle ne sera pas inférieure au nombre de millisecondes spécifié en entrée, soit 500 millisecondes dans le cas qui nous intéresse. Si vous ouvrez un menu local sur l'icône et choisissez **Show»Label**, l'étiquette `Wait Until Next ms Multiple` apparaît à l'écran.

3. Ouvrez un menu local sur l'entrée de la fonction **Wait Until Next ms Multiple** puis choisissez **Create Constant**. Une constante numérique apparaît alors à l'écran qui est automatiquement connectée à la fonction.
4. A l'aide de l'outil Texte, entrez `500`. La constante numérique connectée à la fonction **Wait Until Next ms Multiple** indique un temps d'attente de 500 millisecondes (une demi-seconde), ce qui veut dire que la boucle s'exécutera toutes les demi-secondes.



Vous remarquerez que le VI initialise les registres à décalage avec un nombre aléatoire. Si vous n'initialisez pas le terminal de registre à décalage, il prend la valeur par défaut ou la dernière valeur connue de la précédente exécution. Le cas échéant, les premières moyennes ne sont pas significatives.

5. Exécutez le VI et observez ce qui se passe. LabVIEW ne trace que la moyenne sur le graphe.

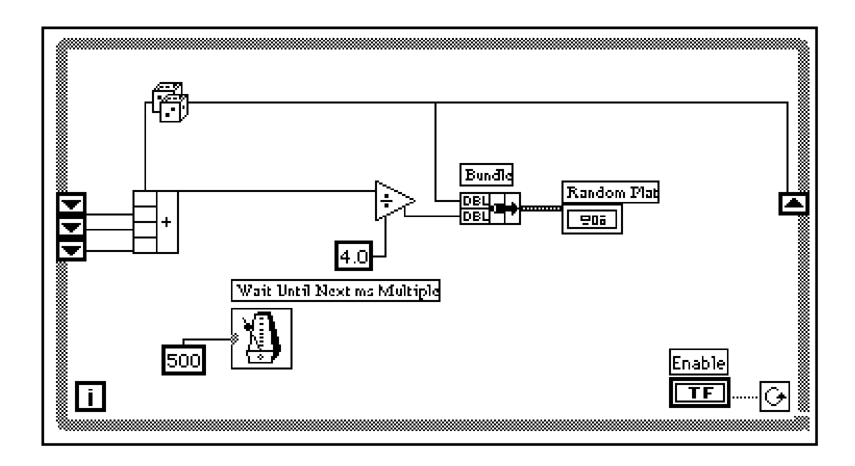


Remarque : ***n'oubliez pas d'initialiser les registres à décalage afin d'éviter d'y intégrer d'anciennes valeurs ou bien encore des valeurs par défaut dans les mesures des données en cours.***

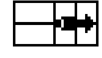
Les graphes déroulants multicourbes

Les graphes déroulants peuvent contenir plusieurs tracés. Lorsqu'il y a plusieurs entrées scalaires, il faut les assembler.

Vous allez modifier le diagramme afin d'afficher la moyenne et la valeur aléatoire courante sur le même graphe déroulant.



1. Modifiez le diagramme conformément à l'illustration précédente.



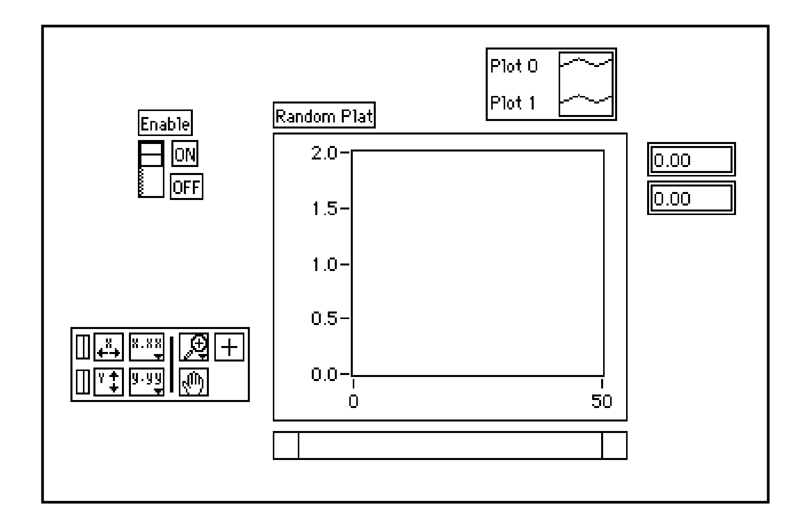
Fonction **Bundle** (**Functions»Cluster**). Dans cet exercice, la fonction **Bundle** *intègre*, ou regroupe, la valeur moyenne et la valeur courante à tracer sur le graphe déroulant. Conformément à l'illustration, le nœud d'assemblage apparaît à gauche lorsque vous le placez dans le diagramme. Si vous ouvrez un menu local sur ce nœud et choisissez **Show»Label**, le mot `Bundle` s'affiche dans l'étiquette. Vous pouvez agrandir ce nœud en lui ajoutant des éléments supplémentaires. Pour ce faire, utilisez le curseur de redimensionnement (que vous obtenez en plaçant l'outil Flèche dans le coin de la fonction).

Remarque : ***l'ordre des données en entrée de la fonction Bundle détermine l'ordre dans lequel sera effectué le tracé des points sur le graphe déroulant. Par exemple, si vous connectez la variable brute à la donnée principale du nœud d'assemblage et la moyenne à la dernière donnée, le premier tracé correspondra aux données brutes et le second à la moyenne.***

2. Exécutez le VI. Le VI affiche deux tracés sur le même graphe déroulant. Les tracés sont superposés, c'est-à-dire qu'ils partagent la même échelle verticale. Essayez d'exécuter le VI en mode Animation pour voir comment se comportent les données dans les registres à décalage. Lorsque vous aurez terminé, n'oubliez pas d'arrêter cette fonction pour que le VI puisse s'exécuter à pleine vitesse.

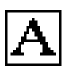
Personnalisation des graphes déroulants

Vous pouvez personnaliser les graphes déroulants pour répondre à des besoins d'affichage particuliers et recevoir des informations complémentaires comme : des barres de défilement, des légendes, des palettes et des afficheurs numériques.



Dans le graphe déroulant, l'afficheur numérique a été activé. Vous remarquerez que chaque tracé du graphe a son propre afficheur numérique.

1. Si la barre de défilement est visible, masquez-la en cliquant dessus pour ouvrir un menu local et désactivez l'option **Show»ScrollBar**.
2. Personnalisez l'axe des Y.
   a. Utilisez l'outil Texte pour double-cliquer sur 2.0 dans l'échelle des Y. Tapez `1.2` puis appuyez sur `<Enter>` (Windows); `<return>` (Macintosh); `<Return>` (Sun); ou `<Enter>` (HP-UX).



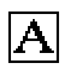
b. Toujours avec l'outil Texte, cliquez sur l'avant-dernier nombre de l'axe des Y. Remplacez ce nombre par 0.2, ou 0.5 par exemple. Ce nombre détermine le pas de graduation de l'axe des Y.



Remarque : ***la taille du graphe a une incidence directe sur la graduation des axes. Agrandissez la taille du graphe déroulant si vous avez des difficultés à personnaliser les axes.***

3. Affichez la légende en ouvrant un menu local sur le graphe déroulant et en choisissant **Show»Legend**. Au besoin, déplacez la légende.

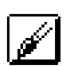






Vous pouvez placer la légende n'importe où dans le graphe déroulant. Agrandissez-la pour y inclure deux tracés en utilisant le curseur de redimensionnement. L'outil Flèche se transforme en curseur de redimensionnement pour indiquer que vous pouvez redimensionner la légende. Remplacez `0` par `Current Value` en double-cliquant sur le texte avec l'outil Texte et en tapant le nouveau texte. Vous pouvez pareillement remplacer le tracé 1 par `Running Avg`. Si le texte disparaît, élargissez le rectangle réservé au texte de légende en le redimensionnant à partir de l'angle *gauche* de la légende avec le curseur de redimensionnement. Vous pouvez aussi définir le style des tracés, avec des lignes et des points, en cliquant sur la légende pour ouvrir un menu local.

Vous pouvez définir l'épaisseur du tracé en ouvrant un menu local dans la légende. En utilisant ce menu local, vous pouvez modifier les paramètres par défaut de la ligne en affectant une valeur supérieure à 1 pixel. Vous avez aussi la possibilité de sélectionner une épaisseur égale à celle d'un cheveu, invisible à l'écran mais qui est en revanche imprimée si votre imprimante supporte l'impression d'une ligne d'une telle finesse.

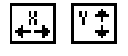


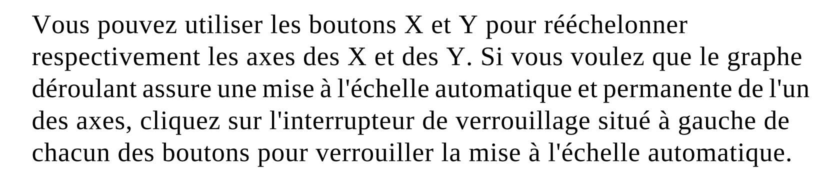
Si vous avez un moniteur couleur, vous pouvez également colorier le fond du graphe et des tracés en cliquant avec l'outil Pinceau sur l'élément que vous souhaitez modifier dans la légende pour faire apparaître un menu local. Choisissez la couleur qui vous convient parmi les couleurs présentées.

4. Affichez la palette locale du graphe déroulant en ouvrant son menu local, et choisissez l'option **Show»Palette**.

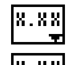
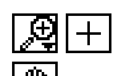
Avec la palette, vous pouvez modifier la présentation du graphe déroulant pendant que le VI s'exécute. Vous pouvez réinitialiser le graphe déroulant, modifier l'échelle de l'axe des X ou des Y et redimensionner l'affichage à tout moment. Vous pouvez également parcourir le contenu d'autres zones ou faire des zooms sur des zones du graphe ou du graphe déroulant. Comme pour la légende, vous pouvez placer la palette n'importe où dans le graphe déroulant.

5. Lancez le VI. Pendant l'exécution du VI, utilisez les boutons de la palette pour modifier le graphe déroulant.





Vous pouvez utiliser les boutons X et Y pour rééchelonner respectivement les axes des X et des Y. Si vous voulez que le graphe déroulant assure une mise à l'échelle automatique et permanente de l'un des axes, cliquez sur l'interrupteur de verrouillage situé à gauche de chacun des boutons pour verrouiller la mise à l'échelle automatique.

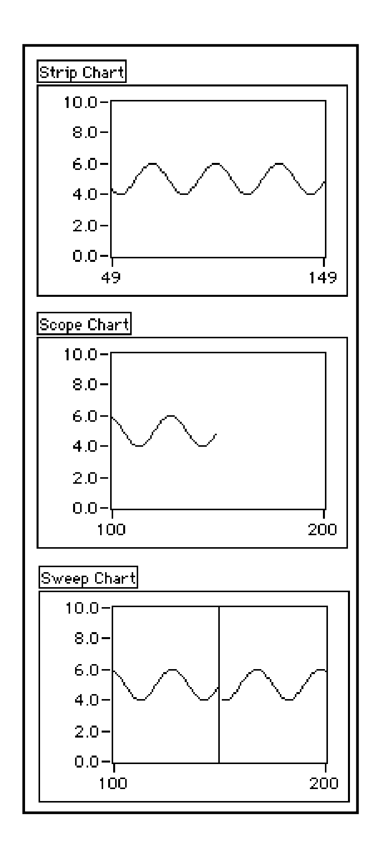
Vous pouvez utiliser les autres boutons pour modifier le contenu du texte sur les axes ou pour contrôler le mode de fonctionnement du graphe déroulant. Exercez-vous à utiliser ces boutons afin de bien assimiler leur fonctionnement, parcourir la zone affichée ou faire des gros plans des zones du graphe déroulant.



Remarque : ***la modification du format du texte des axes nécessite bien souvent plus d'espace par rapport à l'espace initialement prévu pour les axes. Si vous les modifiez, le texte risque de dépasser la taille maximale autorisée. Pour y remédier, utilisez le curseur de redimensionnement pour réduire la zone du graphe déroulant réservée à l'affichage.***

Les différents modes d'affichage

La figure suivante présente les trois modes d'affichage disponibles à partir de la palette **Data Operations»Update Mode** : le graphe déroulant, l'oscillographe et le graphe à balayage. Le mode d'affichage par défaut est le graphe déroulant. (Si le VI est toujours en cours d'exécution, le sous-menu **Data Operations** correspond au menu local du graphe.)



L'affichage en mode *graphe déroulant* s'apparente à un enregistreur papier. En effet, lorsque le VI reçoit une nouvelle valeur, il trace à droite le point correspondant à cette valeur et déplace les anciennes valeurs sur la gauche.

1. Assurez-vous que le VI est toujours en cours d'exécution, ouvrez un menu local sur le graphe déroulant puis choisissez **Data Operations»Update Mode»Scope Chart**.

En mode *oscillographe*, le tracé se rafraîchit comme sur un oscilloscope. Au fur et à mesure que le VI reçoit une nouvelle valeur, il trace le point correspondant à cette valeur à droite du dernier point. Lorsque le tracé atteint la bordure droite de la zone d'affichage de la courbe, le VI efface le tracé et recommence son affichage à partir de la bordure gauche. L'oscillographe est beaucoup plus rapide que le graphe déroulant puisqu'il n'est pas nécessaire de redessiner toute la courbe à l'arrivée de chaque nouveau point (ce qui se traduit par un glissement de l'ensemble de la courbe vers la gauche).

2. Assurez-vous que le VI est toujours en cours d'exécution, ouvrez un menu local sur le graphe déroulant, puis choisissez **Data Operations»Update Mode»Sweep Chart**.

Le mode *graphe à balayage* se comporte sensiblement de la même façon que le mode oscillographe à la différence que l'écran du graphe ne s'efface pas lorsque le point arrive sur la bordure droite. Au contraire, une ligne verticale mobile de séparation se déplace dans la zone d'affichage au fur et à mesure que le VI ajoute de nouvelles données.

3. Arrêtez le VI, et enregistrez-le sous le nom `My Random Average.vi`.

Résumé

LabVIEW est doté de deux structures en boucle pour satisfaire l'exécution répétitive d'un sous-diagramme : la boucle *While* et la boucle *For*. Ces deux structures sont des boîtes réajustables. Vous placez le sous-diagramme à répéter à l'intérieur du cadre de la structure de boucle. La boucle *While* s'exécute tant que la valeur du terminal conditionnel est TRUE. La boucle *For* s'exécute un nombre de fois défini.

Vous pouvez contrôler le temps de cycle de la boucle en utilisant la fonction **Wait Until Next ms Multiple**. Cette fonction garantit que la durée de chaque itération ne sera pas inférieure au nombre exprimé en millisecondes (1000 ms égalent une seconde).

Les registres à décalage (disponibles pour les boucles *While* et *For*) permettent de transférer les valeurs entre la fin de chaque itération et le début de la suivante. Vous avez la possibilité de configurer les registres à décalage pour accéder aux valeurs des itérations précédentes. Pour chaque itération précédente, dont vous souhaitez rappeler une valeur, il vous faut ajouter un nouvel élément sur le terminal de gauche du registre à décalage.

Lorsque LabVIEW doit modifier la représentation d'une variable numérique d'un terminal pour qu'elle corresponde à la représentation de la variable numérique d'un autre terminal, un point de coercition gris apparaît pour symboliser cette conversion. Ce point de coercition apparaît dans le terminal dont la valeur a été convertie.

Quelques informations supplémentaires

Le reste de ce chapitre est consacré à des fonctions plus avancées. N'hésitez pas à consulter dès à présent ces informations, ou bien passez directement au chapitre suivant et revenez à ces fonctions lorsque vous en aurez besoin.

Personnalisation des graphes déroulants

Pour plus d'informations sur les graphes déroulants, veuillez vous reporter au chapitre 15, *Les commandes et les indicateurs de clusters et de tableaux*, du *Manuel de l'utilisateur LabVIEW*.

Accélération des rafraîchissements

Vous pouvez transmettre un tableau à valeurs multiples à un graphe déroulant. Celui-ci traite ces données comme s'il s'agissait de nouvelles données individuelles correspondant à un même tracé. Veuillez vous reporter à l'exemple `charts.vi` qui se trouve dans le répertoire `examples\general\graphs\charts.llb`.

Empilements et superpositions

Dans ce chapitre vous avez créé un graphe multicourbes avec une superposition de points. Vous pouvez aussi empiler des tracés dans un graphe déroulant. Veuillez vous reporter à l'exemple `charts.vi` qui se trouve dans le répertoire `examples\general\graphs\charts.llb`.

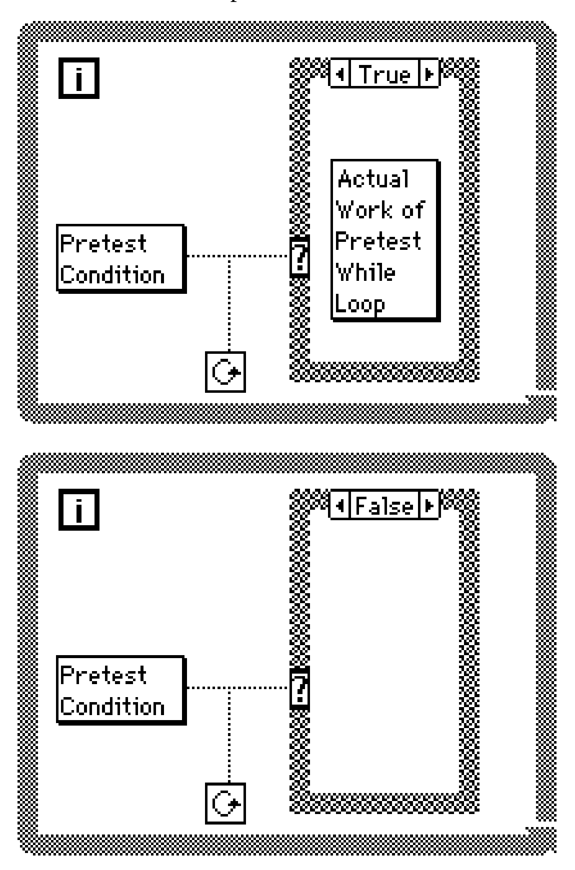
Mise en œuvre des boucles

Les boucles *While* et *For* sont les structures fondamentales de la programmation avec LabVIEW, tant et si bien que vous les retrouverez dans la plupart des exemples et des exercices de ce tutorial. Vous trouverez également des informations supplémentaires sur ces boucles dans le chapitre 19, *Les Structures*, du *Manuel de l'utilisateur LabVIEW*.

Test d'une boucle While avant exécution

Une boucle *While* s'exécute toujours au moins une fois, puisque LabVIEW effectue le test de boucle en continu après chaque exécution du diagramme. Sachez qu'en ajoutant une structure Condition au début de la boucle, vous pouvez construire une boucle *While* qui commence

par tester la condition. Vous câblez le terminal de sélection de la Condition booléenne sur le terminal de sélection de la structure Condition, de sorte que le sous-diagramme correspondant à la condition FALSE s'exécute si la boucle *While* n'est pas censée s'exécuter. Le sous-diagramme correspondant à la condition TRUE contient le diagramme de la boucle *While*. Le test pour continuer ou non se déroule à l'extérieur de la structure Condition et les résultats obtenus sont câblés à la fois sur le terminal conditionnel de la boucle *While* et sur le terminal de sélection de la structure Condition. Dans la figure suivante, les étiquettes représentent le prétest de la condition et le travail réellement effectué par la boucle *While*.



Cet exemple équivaut au pseudo-code suivant :

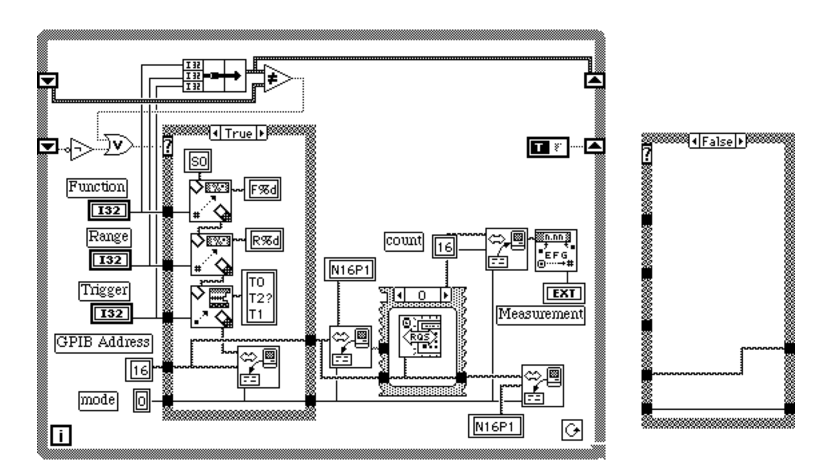
```
Tant que (prétest de la condition)
Effectuer le travail réel de la boucle While
Boucle
```

Mise en œuvre des registres à décalages non initialisés

Vous initialisez un registre à décalage en câblant une variable venant de l'extérieur d'une boucle *While* ou d'une boucle *For* sur le terminal de gauche du registre à décalage. Cependant, vous aurez parfois besoin d'exécuter en continu un VI avec une boucle et un registre à décalage, de manière à ce qu'à chaque exécution du VI la valeur du registre à décalage de l'appel soit conservée. Pour ce faire, vous ne câblerez pas de variable extérieure sur le terminal de gauche du registre à décalage.

Vous pouvez utiliser des registres à décalage non initialisés, par exemple, pour éviter d'avoir à programmer de nouveau les paramètres définissant la fonction, la gamme et le mode de déclenchement d'un driver d'instrument à chaque fois qu'on l'appelle. Cela contribue à l'amélioration de la performance des instruments connus pour leur lenteur à exécuter des commandes.

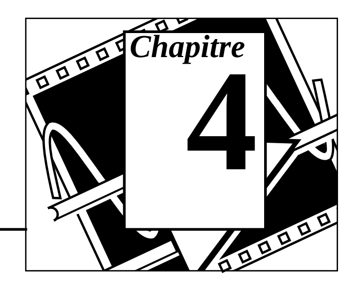
Dans le schéma ci-après, le driver pour le multimètre Fluke 8840A utilise deux registres à décalage non initialisés pour rappeler le dernier état du driver si bien que vous n'avez besoin de reprogrammer l'instrument que lorsque vous l'appelez pour la première fois ou lorsque vous souhaitez modifier un paramètre.



La première fois que ce VI s'exécute, une fois chargé ou compilé, la valeur du registre à décalage du bas est FALSE, soit la valeur par défaut pour un interrupteur tout-ou-rien non initialisé. La structure Condition s'exécute et règle les paramètres de fonction, de gamme et de déclenchement. Il définit également le vu-mètre à utiliser en mode de mesure lent. La condition TRUE s’exécute également si la valeur d’un des trois paramètres diffère par rapport à la valeur précédente, à savoir, lorsque l’une des commandes est modifiée. Vous pouvez modifier le fonctionnement et utiliser une structure Condition pour chaque commande.

Les structures Condition sont reprises en détail dans le chapitre 5, *Structures Condition, structures Séquence et boîte de calcul*.

Tableaux, clusters et graphes

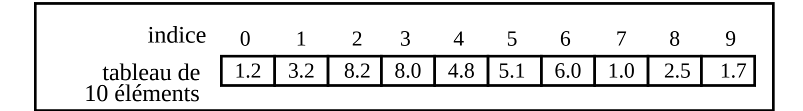


Vous allez apprendre :

- Ce que sont les tableaux.
- Comment générer des tableaux sur le pourtour des boucles.
- Ce qu'est le polymorphisme.
- Ce que sont les *clusters*.
- Comment utiliser les graphes déroulants pour afficher les données.
- Comment utiliser quelques-unes des fonctions de base dans les tableaux.

Les tableaux

Un tableau se définit comme un ensemble de données du même type. Il peut être à une ou plusieurs dimensions pouvant comporter jusqu'à $2^{31} - 1$ éléments à condition bien sûr que la capacité mémoire le permette. Dans LabVIEW, les tableaux peuvent comporter tout type d'élément (à l'exception des tableaux, graphes déroulants ou graphes). Vous accédez à chaque élément du tableau grâce à son indice. L'indice est un chiffre compris entre 0 et *n*-1, où *n* correspond au nombre d'éléments contenus dans le tableau. Le tableau de valeurs numériques reproduit ci-après est un exemple de tableau à une dimension. Vous remarquerez que l'indice du premier élément est égal à 0, celui du deuxième à 1, et ainsi de suite.



Les commandes, constantes et indicateurs de tableaux

Vous créez des commandes, constantes et indicateurs de tableau sur la face-avant ou dans le diagramme en associant une *constante de tableau* à une valeur numérique, booléenne, une chaîne de caractères ou un

cluster. Seule restriction : un élément du tableau ne peut pas être un autre tableau, un graphe ou un graphe déroulant.

Pour des exemples de tableaux, veuillez vous reporter au répertoire `examples\general\arrays.llb`.

Les graphes

Un *indicateur de graphe* est un affichage à deux dimensions d'un ou de plusieurs tableaux de données que l'on appelle *tracés*. LabVIEW propose trois types de graphes différents : les *graphes XY* , les *graphes oscilloscopiques* et les *graphes d'intensité* (veuillez vous reporter à la section *Quelques informations supplémentaires* à la fin de ce chapitre pour en savoir plus sur les graphes d'intensité).

La différence entre un graphe et un graphe déroulant (point abordé dans le chapitre 3, *Boucles et graphes déroulants*, de ce tutorial) réside dans le fait que le premier trace des données en bloc tandis que le second trace les données point par point ou tableau par tableau.

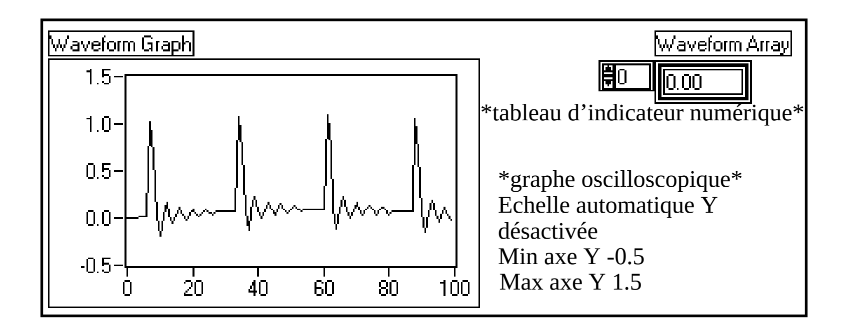
Pour des exemples de VIs de graphes, veuillez vous reporter au répertoire `examples\general\graphs`.

Création d'un tableau par auto-indexation

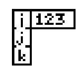
OBJECTIF Créer un tableau en utilisant la fonction d'auto-indexation d'une boucle *For* et tracer le tableau sous la forme d'un graphe oscilloscopique.

Vous allez construire un VI qui va d'abord générer un tableau en utilisant le VI **Generate Waveform** puis tracer le tableau sous la forme d'un graphe oscilloscopique. Vous allez également modifier le VI pour obtenir plusieurs tracés.

La face-avant



1. Ouvrez une nouvelle face-avant.

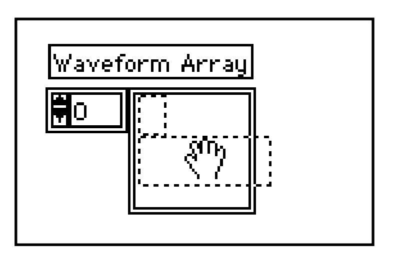


2. Sélectionnez un tableau vierge dans **Controls»Array & Cluster** et placez-le dans la face-avant. Donnez-lui l'étiquette `Waveform Array`.





3. Sélectionnez un indicateur numérique dans **Controls»Numeric** et positionnez-le dans la zone d'affichage des éléments du tableau, conformément à l'illustration suivante. Cet indicateur affiche le contenu du tableau.



Conformément à ce que nous avons vu précédemment, un *indicateur de graphe* est un affichage à deux dimensions d'un ou de plusieurs tableaux de données appelés *tracés*. LabVIEW offre trois types de graphes : les *graphes XY*, les *graphes oscilloscopiques* et les *graphes d'intensité*.

4. Sélectionnez un graphe oscilloscopique dans **Controls»Graph** et placez-le dans la face-avant. Donnez-lui l'étiquette `Waveform Graph`.

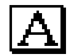
Ce graphe oscilloscopique trace des tableaux avec des points à intervalle régulier, comme dans le cas des courbes variant avec le temps.

5. Agrandissez ce graphe en faisant glisser l'un de ses coins à l'aide du curseur de redimensionnement.

Par défaut, les graphes *mettent automatiquement à l'échelle* leurs entrées. C'est-à-dire qu'ils ajustent automatiquement les limites des axes X et Y pour visualiser l'intégralité des données.

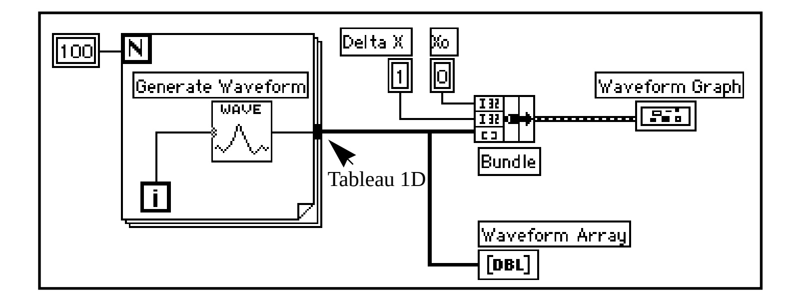
6. Désactivez la fonction de mise à l'échelle automatique. Pour ce faire, ouvrez un menu local sur le graphe et désactivez **Y Scale»Autoscale Y**.

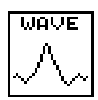


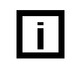


7. Modifiez les limites de l'axe Y en double-cliquant dessus à l'aide de l'outil Texte et en entrant les nouvelles valeurs. Tapez pour la valeur la plus basse de l'axe des Y -0.5 et pour la valeur la plus haute 1.5.

Le diagramme







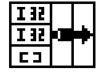
1. Construisez le diagramme reproduit ci-dessus.

Le VI **Generate Waveform** (**Functions»Tutorial**) restitue un point d'une courbe. Ce VI nécessite une entrée d'indice scalaire ; câblez le terminal d'itération de la boucle à cette entrée. Le fait d'ouvrir un menu local sur le VI et de sélectionner **Show»Label** affiche le mot `Generate Waveform` dans l'étiquette.

Vous remarquerez que le fil du VI **Generate Waveform** s'épaissit en se transformant en tableau à la bordure de la boucle.

La boucle *For* empile automatiquement les tableaux sur son pourtour. Ce phénomène s'appelle l'*auto-indexation*. Dans ce cas, la constante

numérique 100 câblée à l'entrée numérique de comptage de la boucle permet à la boucle *For* de créer un tableau de 100 éléments (indexés de 0 à 99).



La fonction **Bundle** (**Functions»Cluster**) rassemble les composantes du tracé en un *cluster*. Vous devez redimensionner l'icône de la fonction **Bundle** pour pouvoir la câbler correctement. Pour ce faire, positionnez l'outil Flèche en bas à droite de l'icône. L'outil se transforme alors en curseur de redimensionnement. A ce moment-là, cliquez dessus et faites-le glisser jusqu'à ce qu'une troisième broche apparaisse. Vous pouvez alors continuer à câbler le diagramme en vous inspirant de l'illustration précédente.

Un *cluster* peut contenir des éléments de types différents. Par exemple, dans le diagramme que vous êtes en train de construire, le *cluster* regroupe des éléments provenant de plusieurs endroits différents du diagramme pour limiter l'enchevêtrement du câblage. Lorsque vous utilisez des *clusters*, il vous faut moins de broches pour câbler vos sous-VIs. Un *cluster* s'apparente à un enregistrement en Pascal ou à une structure en C. Vous pouvez le comparer à un bouquet de fils, un peu comme ceux des centraux téléphoniques, où chaque fil correspond à un élément différent du *cluster*. Les composantes de ce *cluster* comprennent la valeur initiale de X (0), la valeur de delta-X (1) et le tableau des Y (données de la courbe, fournies par les constantes numériques du diagramme). Dans LabVIEW, utilisez la fonction **Bundle** pour former un *cluster*.



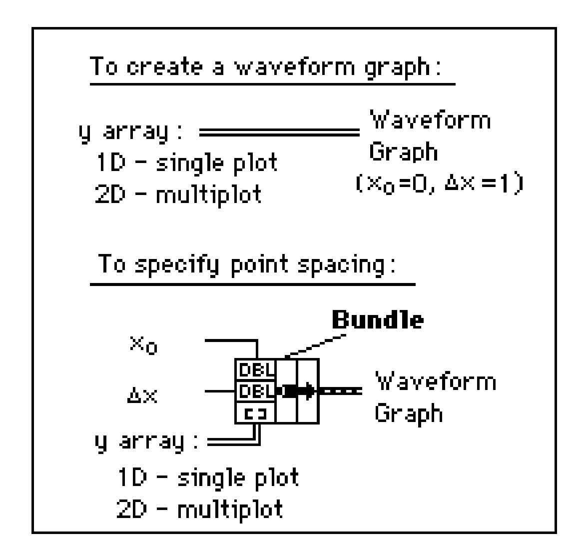
Remarque : ***assurez-vous de ne construire que des types de données reconnus par les graphes et les graphes déroulants.***

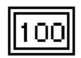
Au fur et à mesure que vous construisez votre diagramme, vérifiez bien les types de données en procédant de la manière suivante :

- Ouvrez la fenêtre d'aide en choisissant **Help»Show Help**.



- Déplacez l'outil Bobine au-dessus du terminal du graphe.
- Vérifiez l'exactitude des informations qui apparaissent dans la fenêtre d'aide. A titre d'exemple, reportez-vous à l'illustration suivante.





Numeric Constant (**Functions»Numeric**). Trois constantes numériques permettent de fixer le nombre d'itérations de la boucle *For*, la valeur initiale de X et la valeur de delta-X. Vous remarquerez que vous avez toujours la possibilité d'ouvrir un menu local sur le terminal de comptage de la boucle *For*, présenté à gauche, puis de sélectionner **Create Constant** pour ajouter et câbler automatiquement une constante numérique à ce terminal.



Chaque itération de la boucle *For* génère un point de la courbe que le VI stocke dans le tableau de courbe créé automatiquement sur la bordure de la boucle. Une fois que la boucle a terminé de s'exécuter, la fonction **Bundle** rassemble la valeur initiale de X (Xo), la valeur de delta-X et le tableau pour les tracer sur le graphe.

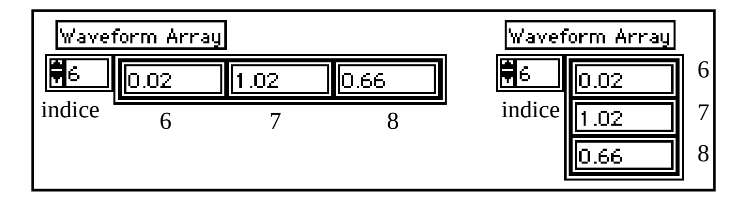
2. Revenez dans la face-avant et exécutez le VI. Il trace le tableau auto-indexé de la courbe dans le graphe déroulant. La valeur initiale de X est égale à 0 et la valeur de delta-X est égale à 1.
3. Modifiez la valeur de delta-X en 0.5 et la valeur initiale de X en 20. Lancez de nouveau le VI.

Vous remarquerez que le graphe affiche maintenant les mêmes 100 points de données avec une valeur de départ de 20 et un delta-X de 0.5 entre chaque point (voir l'axe des X). Dans un test de synchronisation, le graphe correspondrait à un enregistrement de 50 secondes à 20

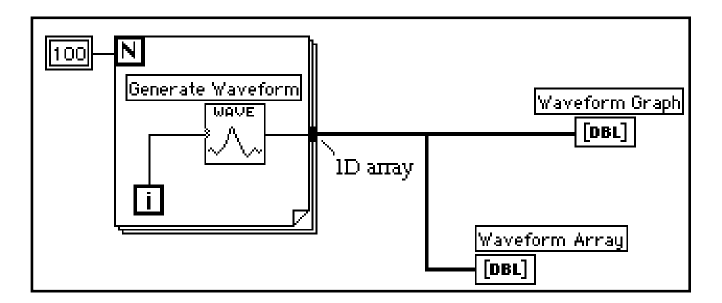
secondes du début. Exercez-vous en modifiant la valeur initiale et le delta-X.

4. Vous avez la possibilité de visualiser n'importe quel élément du tableau en entrant l'indice de cet élément dans l'affichage d'indice. Si vous saisissez un nombre supérieur à la taille du tableau, l'afficheur s'assombrit, indiquant par là qu'aucune valeur n'a été définie pour cet indice.

Si vous souhaitez visualiser plusieurs éléments à la fois, vous pouvez redimensionner l'indicateur du tableau. Positionnez l'outil Flèche en bas à droite du tableau. L'outil se transforme alors en curseur de redimensionnement. Lorsque l'outil se transforme, faites-le glisser vers la droite ou vers le bas. Le tableau affiche alors plusieurs éléments dans l'ordre croissant d'indexation en commençant par l'élément correspondant à l'indice spécifié, conformément à l'illustration suivante.



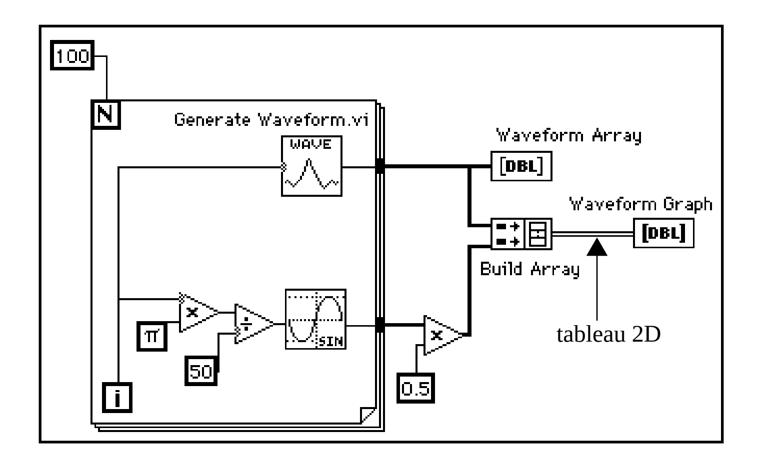
Dans le diagramme précédent, vous avez indiqué une valeur initiale de X et une valeur de delta-X pour la courbe. Sachez qu'il est fréquent que la valeur initiale de X soit égale à zéro et que la valeur de delta-X soit égale à 1. Dans ces cas-là, vous pouvez câbler le tableau de la courbe directement au terminal du graphe oscilloscopique comme le montre l'illustration ci-après.



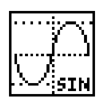
5. Revenez dans la fenêtre du diagramme. Effacez la fonction **Bundle** ainsi que les constantes numériques qui lui sont affectées. Pour ce faire, choisissez la fonction et les constantes à l'aide de l'outil Flèche, puis appuyez sur la touche `<Suppr>`. Choisissez **Edit»Remove Bad Wires**. Finissez de câbler le diagramme comme indiqué dans l'illustration précédente.
6. Lancez le VI. Vous remarquerez qu'il trace la courbe avec une valeur initiale de X égale à 0 et une valeur de delta-X égale à 1.

Les graphes multicourbes

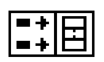
Vous pouvez créer des graphes multicourbes en construisant un tableau avec des données du même type que celles d'un graphe à un tracé unique.



1. Poursuivez la construction de votre diagramme jusqu'à ce qu'il ressemble au diagramme de la figure précédente.



Fonction **Sine** (**Functions»Numeric»Trigonometric**). Dans cet exercice, cette fonction, utilisée dans une boucle *For*, va vous permettre de construire un tableau de points représentant une période de sinusoïde.



Fonction **Build Array** (**Functions»Array**). Dans cet exercice, cette fonction va vous permettre de créer la structure de données qui convient le mieux pour tracer deux tableaux sur un graphe oscilloscopique, à savoir, un tableau à deux dimensions. Agrandissez

la fonction **Build Array** pour créer deux entrées en tirant sur un coin à l'aide de l'outil Flèche.

Constante **Pi** (**Functions»Numeric»Additional Numeric Constants**).

Petit rappel : les fonctions **Multiply** et **Divide** se trouvent dans la palette **Functions»Numeric**.

2. Revenez dans la face-avant puis lancez le VI.

Vous remarquerez que les deux courbes sont bien tracées dans le même graphe oscilloscopique. La valeur initiale de X est prise par défaut, soit 0, de même que la valeur de delta-X, soit 1, pour les deux ensembles de données.

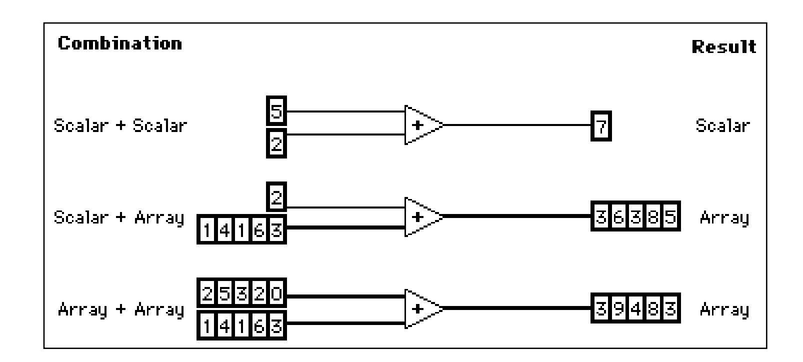
Remarque : ***vous avez la possibilité de modifier le style du graphe. Pour ce faire, ouvrez un menu local en cliquant dans la légende correspondant au tracé voulu. Vous pouvez par exemple obtenir un graphe sous la forme de barres à la place de lignes, en choisissant les options*** **Common Plots»Bar Graph.**

3. Enregistrez et fermez le VI. Nommez-le `My Graph Waveform Arrays.vi`. Assurez-vous que vous avez bien enregistré votre travail dans le répertoire `mywork.llb`.

Le polymorphisme

On appelle polymorphisme la capacité que possède une fonction à s'adapter à des données d'entrée de types, dimensions ou représentations différents. La plupart des fonctions de LabVIEW sont dotées de cette capacité. Le diagramme précédent est un exemple de polymorphisme. Vous remarquerez que la fonction **Multiply** est utilisée à deux endroits, à l'intérieur et à l'extérieur de la boucle *For*. A l'intérieur de la boucle *For*, la fonction multiplie deux valeurs scalaires. A l'extérieur, elle multiplie un tableau par une valeur scalaire.

L'exemple suivant présente quelques-unes des combinaisons polymorphes de la fonction **Add**.



Dans la première combinaison, les deux valeurs scalaires sont additionnées, le résultat obtenu étant une autre valeur scalaire. Dans le deuxième cas de figure, la valeur scalaire est ajoutée séparément à chaque élément du tableau, ce qui donne un nouveau tableau. Dans la troisième combinaison, chaque élément du tableau est ajouté à l'élément correspondant d'un autre tableau. Vous avez également la possibilité d'utiliser d'autres combinaisons, comme les *clusters* de numériques, les tableaux de *clusters*, et ainsi de suite.

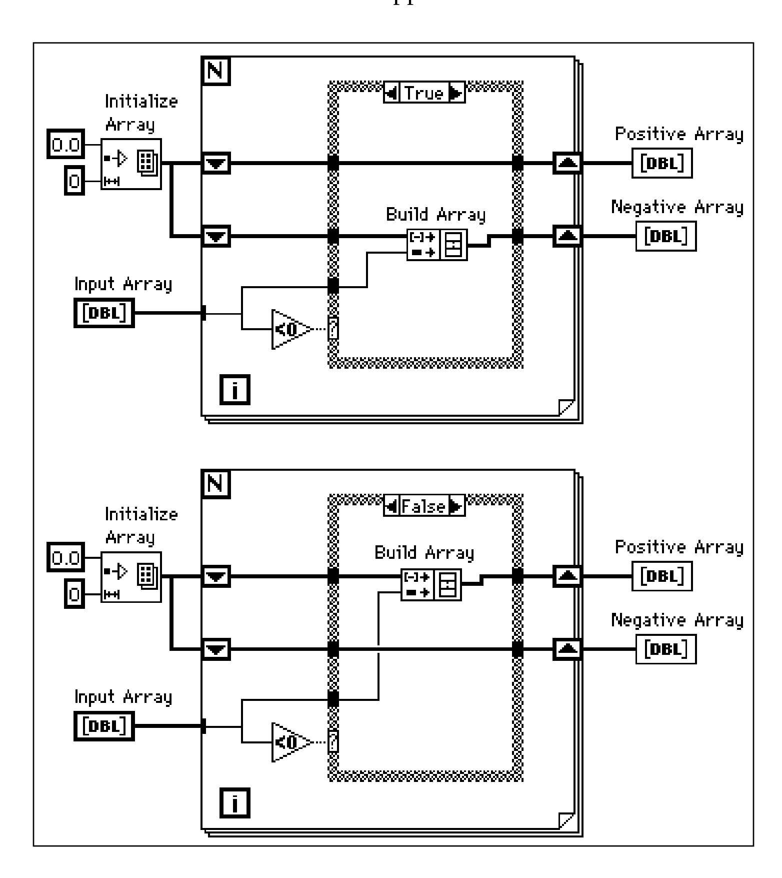
Ces principes peuvent également s'appliquer à d'autres types de données et de fonctions de LabVIEW. Les fonctions LabVIEW sont chacune plus ou moins polymorphes. Certaines acceptent des entrées numériques et booléennes, d'autres une combinaison de n'importe quel type de données. Pour plus d'informations sur le polymorphisme, consultez **Online Reference»Function and VI Reference»Introduction to Functions**.

Auto-indexation des tableaux d'entrée

OBJECTIVE Ouvrir et faire fonctionner un VI qui utilise l'auto-indexation dans une boucle *For* pour travailler sur un tableau.

1. Ouvrez `Separate Array Values.vi` en choisissant **File»Open...**. Le VI se trouve dans le répertoire `examples\general\arrays.llb`.
2. Ouvrez la fenêtre du diagramme. Vous pouvez aussi ouvrir un menu local en cliquant sur le tableau et en choisissant les options **Show Case True** ou **Show Case False** pour visualiser les cas vrais

et faux du tableau. L'illustration suivante présente un diagramme avec les cas TRUE et FALSE apparents.



Vous remarquerez que le câble qui part de `Input Array` change d'épaisseur à l'extérieur de la boucle *For*, indiquant par là qu'il véhicule un tableau. Il s'affine à l'intérieur de la boucle, indiquant par là qu'il s'agit d'un élément unique. Le *i*^{ème} élément du tableau est automatiquement indexé à chaque itération.

Mise en œuvre de l'auto-indexation pour comptage en boucle *For*

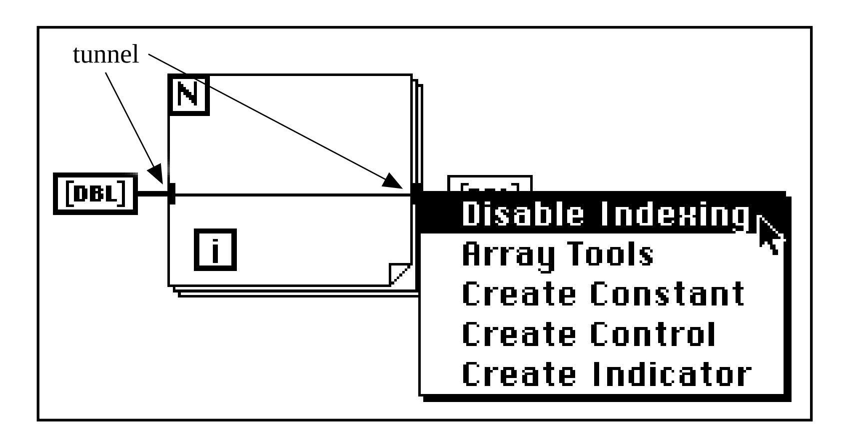


Vous remarquerez que le terminal de comptage n'est pas câblé. Lorsque vous utilisez une auto-indexation sur un tableau *en entrée* d'une boucle *For*, LabVIEW adapte automatiquement le comptage à la taille du tableau. Ce faisant, il n'est pas nécessaire de câbler une valeur au terminal de comptage. Si vous utilisez une auto-indexation sur plus

d'un tableau, ou si vous définissez le comptage, ce dernier prend la plus petite des valeurs possibles.

1. Lancez le VI et testez-le avec différentes tailles de tableaux. Créez une commande numérique sur la face-avant puis câblez-la au terminal de comptage et testez les tableaux de sortie pour voir comment les différents comptages affectent les tableaux en sortie.
2. Fermez le VI sans enregistrer les modifications. Si vous n'êtes pas familier des structures utilisées dans cet exemple, celles-ci sont abordées plus en détail dans la suite de ce tutorial.

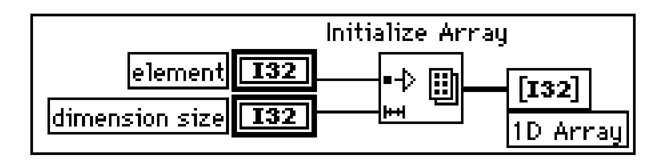
☞ **Remarque :** ***LabVIEW active automatiquement l'auto-indexation pour tous les tableaux câblés à une boucle For. Vous avez la possibilité de désactiver l'auto-indexation en ouvrant un menu local sur le tunnel, c'est-à-dire, le point d'entrée du tableau d'entrée, et en choisissant la fonction*** **Disable Indexing.**



LabVIEW désactive automatiquement l'auto-indexation pour tous les tableaux câblés à une boucle *While*. Il suffit d'ouvrir un menu local sur le tunnel du tableau d'une boucle *While* pour activer l'auto-indexation.

Mise en œuvre de la fonction Initialize Array

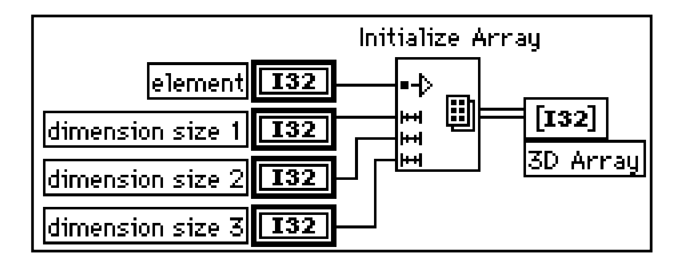
Vous pourrez remarquer que les deux registres à décalage sont initialisés avec la fonction **Initialize Array** qui se trouve dans la palette **Functions»Array**. Utilisez cette fonction pour créer un tableau comportant des éléments de même valeur. Dans l'illustration suivante, cette fonction crée un tableau à une dimension.



L'entrée `element` détermine le type de données et la valeur de chaque élément. L'entrée `dimension size` détermine la longueur du tableau. Par exemple, si `element` est un entier avec la valeur cinq et si `dimension size` a une valeur de 100, nous obtiendrons un tableau à une dimension avec 100 entiers ayant tous comme valeur cinq. Vous pouvez câbler les entrées à partir des terminaux de commande de la face-avant comme dans le cas présent, à partir de constantes d'un diagramme ou bien encore à partir de calculs effectués dans d'autres parties du diagramme.

Pour créer et initialiser un tableau à plus d'une dimension, ouvrez un menu local en cliquant sur le côté inférieur gauche de la fonction puis en choisissant **Add Dimension**. Vous pouvez également utiliser le curseur de redimensionnement pour agrandir la taille du nœud **Initialize Array** et y ajouter d'autres entrées `dimension size`, une entrée pour chaque dimension souhaitée. Vous pouvez supprimer les dimensions en réduisant la taille du nœud et en choisissant l'option **Remove Dimension** dans le menu local ou bien à l'aide du curseur de redimensionnement.

Le diagramme suivant vous montre comment initialiser un tableau à trois dimensions.



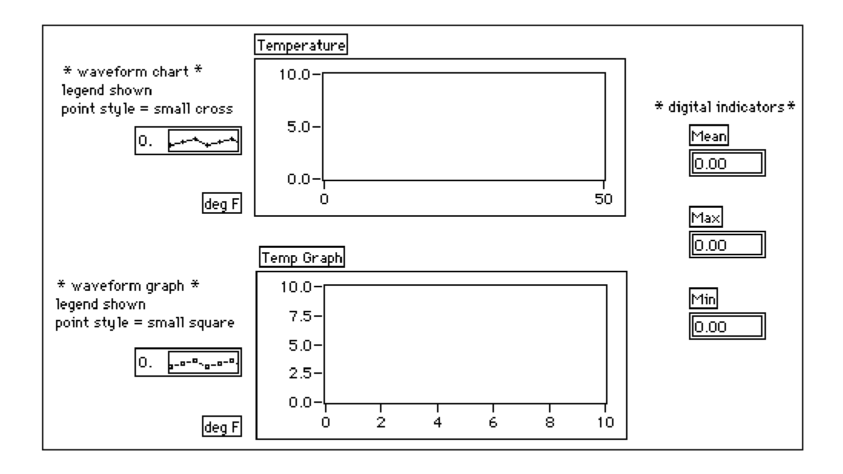
Comme vous avez pu le constater dans l'exemple précédent, si toutes les entrées `dimension size` sont égales à zéro, la fonction crée un tableau vide avec le nombre de dimensions et le type de données voulus.

Mise en œuvre des VIs d'analyse et de graphes

OBJECTIF Créer un VI qui mesure une température toutes les 0,25 secondes pendant 10 secondes. En cours d'acquisition, le VI visualise les mesures en temps réel sur un graphe déroulant. Lorsque le processus d'acquisition est terminé, le VI trace un graphe mettant en surbrillance les températures moyenne, maximale et minimale.

Pour des exemples de VIs d'analyse, reportez-vous au répertoire `examples\analysis`.

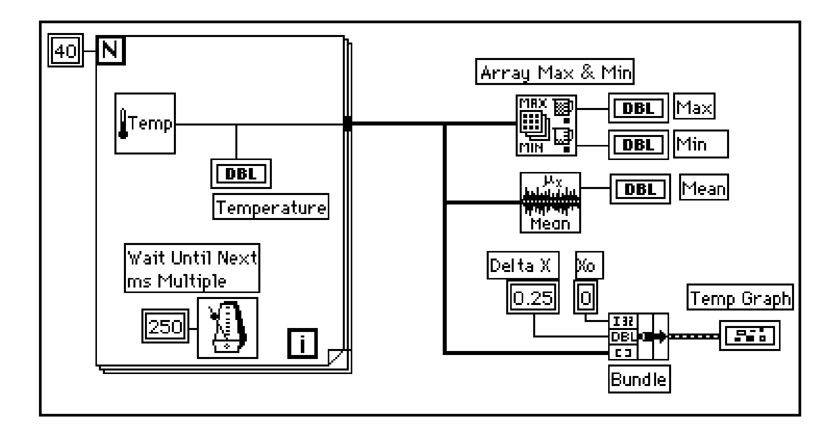
La face-avant



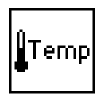
1. Ouvrez une nouvelle fenêtre puis construisez la face-avant reproduite dans l'illustration précédente. Vous pouvez modifier les styles des points des graphes déroulants et des graphes oscilloscopiques en ouvrant un menu local sur leurs légendes.

Le graphe déroulant `Temperature` affichera l'évolution de la température au fur et à mesure de l'acquisition. En fin d'acquisition, le VI tracera les données dans `Temp Graph`. Les indicateurs numériques `Mean`, `Max` et `Min` affichent les températures moyenne, maximale et minimale.

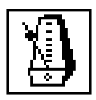
Le diagramme



1. Construisez le diagramme de l'illustration précédente à l'aide des éléments suivants :



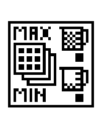
Vous pouvez utiliser le VI **Digital Thermometer** (**Functions»Tutorial**), ou bien encore le VI que vous avez construit au chapitre 2 en choisissant **Functions»Select a VI...** et en sélectionnant `My Thermometer VI.` Celui-ci effectue un relevé de température.



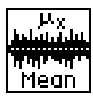
Fonction **Wait Until Next ms Multiple** (**Functions»Time & Dialog**). Dans cet exercice, cette fonction garantit que la boucle *For* s'exécutera toutes les 0,25 secondes (250 millisecondes).



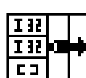
Numeric Constant (**Functions»Numeric**). Vous avez également la possibilité d'ouvrir un menu local sur la fonction **Wait Until Next ms Multiple** et de choisir **Create Constant** afin de créer et de câbler automatiquement la constante numérique.



Fonction **Array Max & Min** (**Functions»Array**). Dans cet exercice, cette fonction permet de retenir les minimum et maximum des températures relevées au cours de l'acquisition.



Le VI **Mean** (**Functions»Analysis»Probability and Statistics**) permet d'obtenir la moyenne des relevés de température.



Fonction **Bundle** (**Functions»Cluster**). Elle rassemble les différents éléments du tracé dans un *cluster*. Parmi les éléments en question, figurent la valeur initiale de X, égale à 0, la valeur de delta-X, égale à

0,25, et le tableau Y des données de température. Utilisez l'outil Flèche pour redimensionner la fonction en tirant sur l'un de ses coins.

La boucle *For* s'exécute 40 fois de suite. La fonction **Wait Until Next ms Multiple** déclenche une itération toutes les 250 millisecondes. Le VI conserve les relevés de température dans chacun des tableaux créés en bordure de la boucle *For* (auto-indexation). Lorsque l'exécution de la boucle *For* est terminée, le tableau exécute plusieurs nœuds.

La fonction **Array Max & Min** retourne les températures minimale et maximale. Le VI retourne la moyenne des relevés de température.

Le VI complet regroupe les données dans un tableau en prenant comme valeur initiale de X, 0 et comme valeur de delta-X, 0.25. Cette valeur de delta-X égale à 0.25 est nécessaire pour que le VI trace sous la forme d'un graphe oscilloscopique l'évolution de la température toutes les 0.25 secondes.

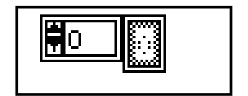
2. Revenez dans la face-avant et exécutez le VI.
3. Enregistrez le VI dans `mywork.llb` sous le nom de `My Temperature Analysis.vi`.

Mise en œuvre des tableaux

LabVIEW offre de nombreuses fonctions permettant de manipuler les tableaux dans la palette **Functions»Array**. Quelques-unes des fonctions les plus courantes sont abordées ici.

Création et initialisation des tableaux

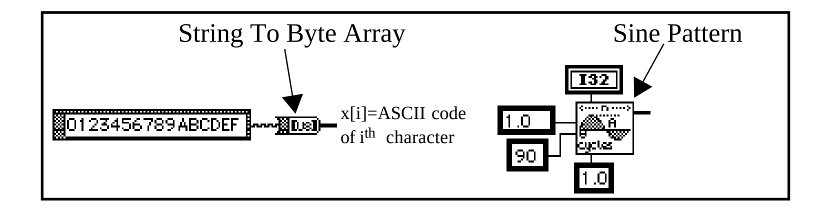
Si vous avez besoin d'un tableau comme source de données dans votre diagramme, vous pouvez choisir la fonction **Functions»Array** pour y sélectionner une constante tableau que vous positionnerez dans votre diagramme. Vous pouvez aussi choisir à l'aide de l'outil Doigt une constante numérique, booléenne ou chaîne de caractères que vous placerez dans le tableau vide. L'illustration suivante reprend un exemple de constante tableau avec une constante numérique intégrée dans le tableau vierge.



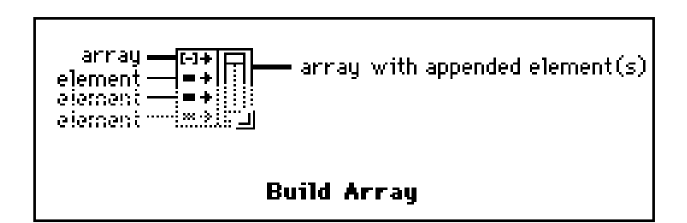
☞ **Remarque :** ***vous pouvez aussi créer un tableau et ses commandes sur la face-avant puis copier ou faire glisser la commande de tableau à l'intérieur du diagramme pour reproduire la constante correspondante.***

Pour plus d'informations sur la manière de créer des commandes et des indicateurs de tableaux sur la face-avant, veuillez vous reporter au chapitre 15, *Les commandes et les indicateurs de clusters et de tableaux*, du *Manuel de l'utilisateur LabVIEW*.

Il existe plusieurs façons de créer et d'initialiser des tableaux dans le diagramme. Vous savez déjà comment créer des tableaux au pourtour des boucles et comment utiliser la fonction **Initialize Array**. Certaines fonctions du diagramme permettent aussi de produire des tableaux comme le montre l'illustration suivante.



Mise en œuvre de la fonction Build Array

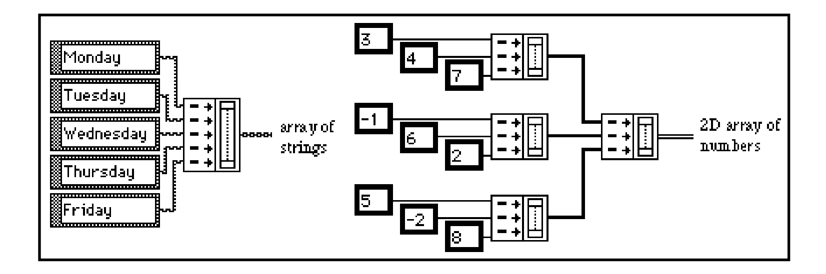


Fonction **Build Array** (**Functions»Array**). Elle permet de créer un tableau à partir de données scalaires ou d'éléments issus d'autres tableaux. Au départ, la fonction **Build Array** ne compte qu'une seule entrée scalaire.

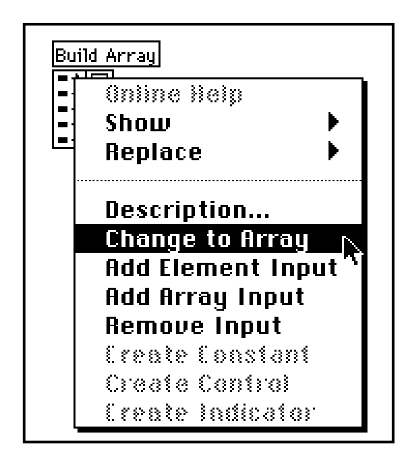
Vous pouvez ajouter autant d'entrées que vous le souhaitez, chaque entrée pouvant être une valeur scalaire ou un tableau. Pour ce faire, ouvrez un menu local sur le côté gauche de la fonction puis choisissez

Add Element Input ou **Add Array Input**. Vous avez également la possibilité d'agrandir le nœud **Build Array** en utilisant le curseur de redimensionnement (placez l'outil Flèche sur le coin d'un objet pour le transformer en curseur de redimensionnement). Vous pouvez également supprimer des entrées en réduisant le nœud à l'aide du curseur de redimensionnement ou en choisissant l'option **Remove Input**.

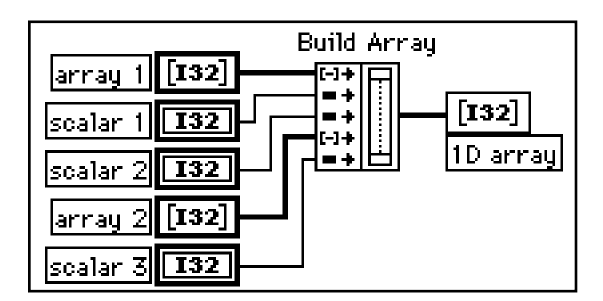
Les exemples suivants vous présentent deux façons de créer et d'initialiser des tableaux avec des constantes. A gauche, cinq constantes chaînes de caractères constituent un tableau à une dimension. A droite, trois groupes de constantes numériques sont regroupés dans trois tableaux numériques à une seule dimension. Les trois tableaux sont ensuite rassemblés en un seul tableau numérique à deux dimensions. Au final, vous obtenez un tableau 3 x 3 avec les lignes suivantes : 3, 4, 7 ; -1, 6, 2 ; et 5, -2 et 8.



Vous pouvez également créer un tableau en combinant d'autres tableaux à des éléments scalaires. Supposons, par exemple, que vous ayez deux tableaux et trois éléments scalaires que vous vouliez combiner dans un nouveau tableau dans l'ordre suivant : tableau 1, scalaire 1, scalaire 2 tableau 2 et scalaire 3. Commencez par créer un nœud **Build Array** avec cinq entrées. Ouvrez un menu local sur la première entrée, celle du haut, et choisissez **Change to Array**, conformément à l'illustration suivante. Faites de même pour la quatrième entrée et ainsi de suite jusqu'à l'avant-dernière.



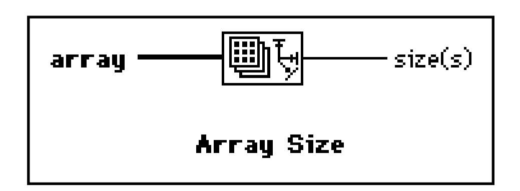
Ensuite, câblez les tableaux et les scalaires au nœud. Le tableau de sortie est un tableau à une dimension composé des éléments du tableau 1, du scalaire 1, du scalaire 2, des éléments du tableau 2 et du scalaire 3, comme illustré ci-dessous.



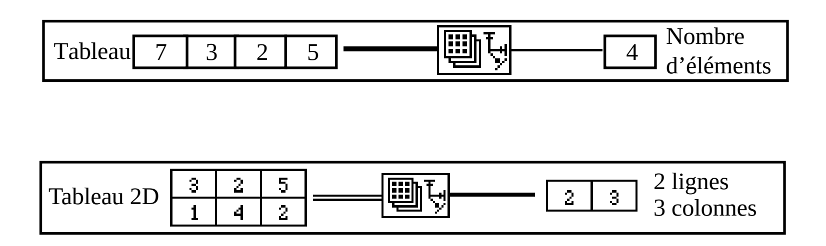
La dimension du tableau de sortie est toujours supérieure d'une unité à celle des éléments que vous câblez aux entrées d'éléments, et égale à celle des tableaux que vous câblez aux entrées de tableaux. Les entrées de tableaux et d'éléments ne peuvent avoir plus d'une dimension de différence. Par exemple, si vous câblez une série de tableaux 1D à des entrées d'éléments, le tableau de sortie sera un tableau 2D dont les lignes correspondront aux entrées 1D. Dans le cas présent, toutes les entrées de tableaux sont des tableaux 2D. Si les entrées d'éléments sont des tableaux 2D, alors la sortie est un tableau 3D et les entrées de tableaux sont des tableaux 3D. Vous ne pouvez pas construire un

tableau avec des scalaires en entrée d'éléments et des tableaux 2D ou plus en entrée de tableaux.

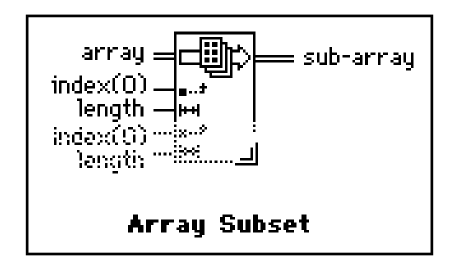
Vérification de la dimension d'un tableau



La fonction **Array Size** donne le nombre d'éléments du tableau en entrée.



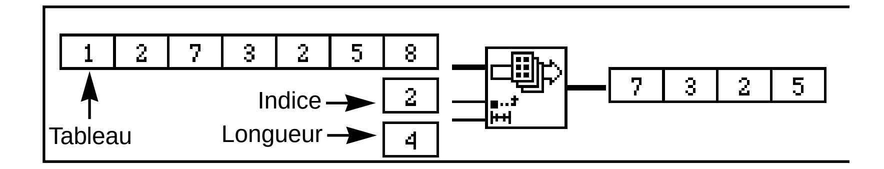
Mise en œuvre de la fonction Array Subset

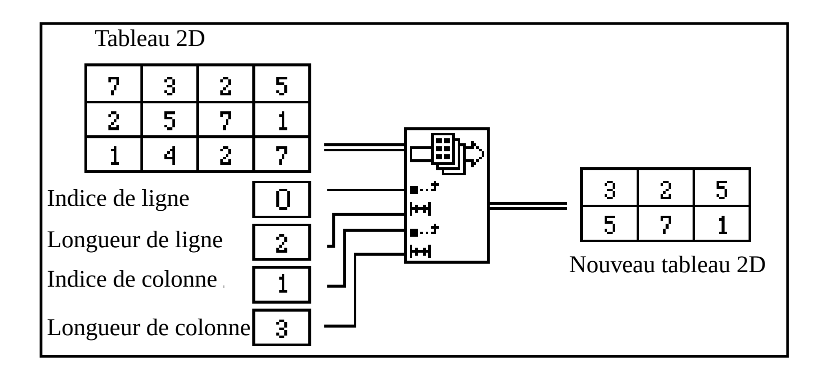


Cette fonction permet d'extraire une partie d'un tableau ou d'une matrice.

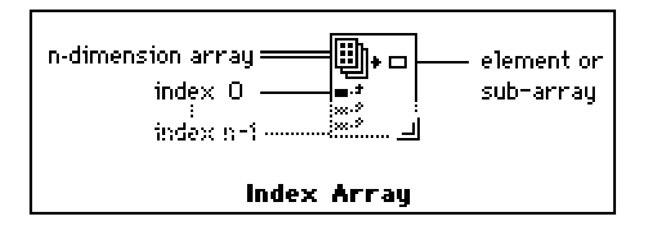
La fonction **Array Subset** restitue une partie d'un tableau, en commençant par **index** (indice du premier élément) et avec **length** (le

nombre d'éléments). Les illustrations suivantes vous fournissent quelques exemples de cette fonction. Vous remarquerez que l'indice du tableau démarre à 0.

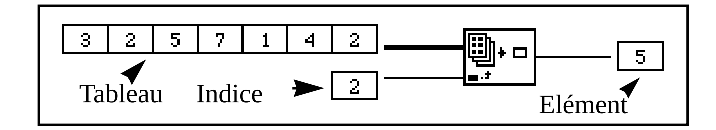




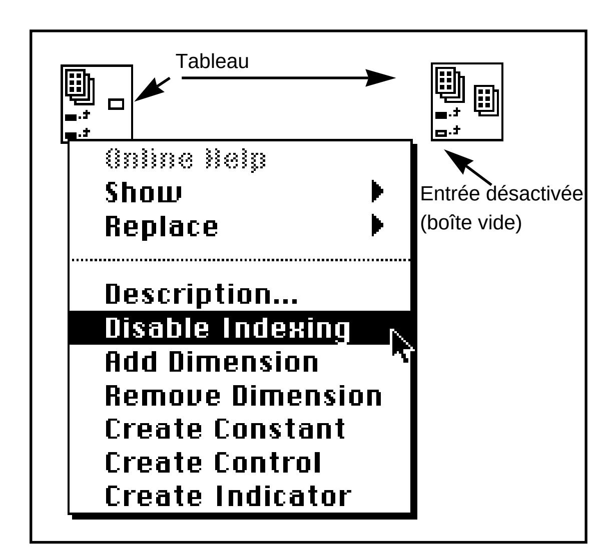
Mise en œuvre de la fonction Index Array



La fonction **Index Array** vous permet d'accéder à un élément particulier d'un tableau. Dans l'exemple suivant, la fonction **Index Array** sert à accéder au troisième élément d'un tableau. Vous remarquerez que l'indice du troisième élément a pour valeur 2 puisque le premier élément a un indice égal à 0.

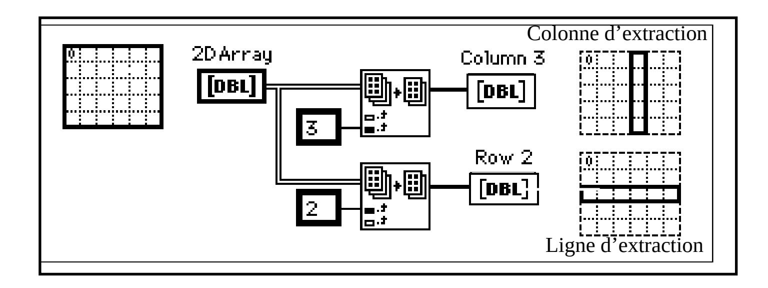


Vous pouvez également utiliser cette fonction pour *découper* une ou plusieurs dimensions d'un tableau multidimensionnel pour produire un sous-tableau de l'original. Pour ce faire, agrandissez la fonction **Index Array** afin d'y inclure deux entrées d'indices, puis choisissez la commande **Disable Indexing** dans le menu local du terminal du second terminal d’indice conformément à l'illustration suivante. Vous venez ainsi de désactiver l'accès à une colonne particulière du tableau. En attribuant ensuite un indice de ligne, vous obtenez un tableau dont les éléments correspondent à ceux d'une ligne particulière du tableau 2D. Vous pouvez tout autant désactiver l'indexation sur le terminal de ligne.

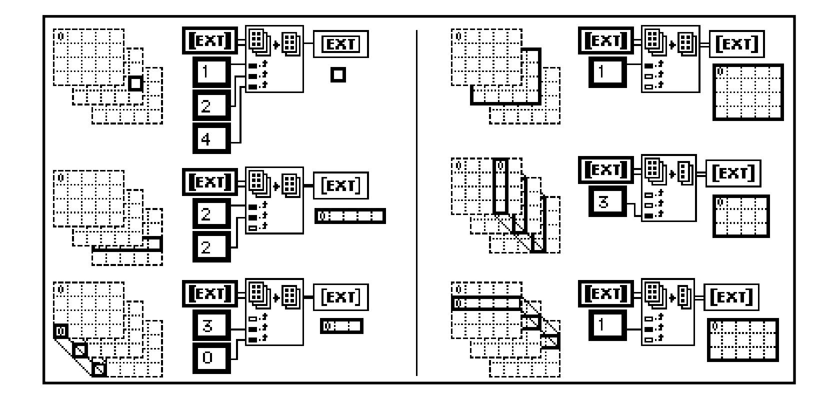


Vous remarquerez que le symbole associé au terminal de l'indice se vide lorsque vous désactivez l'indexation. Pour réactiver un indice, choisissez la commande **Enable Indexing** dans le menu local.

Vous pouvez extraire des sous-tableaux de n'importe quelle combinaison de dimensions. L'exemple suivant vous explique comment extraire des tableaux d'une ligne ou d'une colonne à partir d'un tableau à deux dimensions.



A partir d'un tableau à trois dimensions, vous pouvez extraire un tableau à deux dimensions en désactivant deux terminaux d'indice, ou un tableau à une dimension en désactivant un seul terminal d'indice. L'illustration suivante présente plusieurs façons de découper un tableau à trois dimensions.



Les règles suivantes régissent l'utilisation de la fonction **Index Array** pour découper des tableaux.

1. La dimension de l'objet de sortie doit être égale au nombre de terminaux d'indice désactivés. Par exemple :
    - aucun terminal désactivé égale un élément scalaire ;
    - un terminal désactivé égale un tableau 1D ;
    - deux terminaux désactivés égalent un tableau 2D.

2. Les valeurs câblées aux terminaux activés déterminent les éléments de sortie.

Par conséquent, vous pouvez interpréter l'exemple présenté en bas à gauche comme étant une commande servant à générer un tableau à une dimension composé de tous les éléments de la colonne 0 et de la ligne 3. De même, l'exemple en haut à droite correspond à une commande permettant de générer un tableau à deux dimensions de la page 1. Le nouvel élément $0^{\text{ème}}$ se rapproche le plus de l'original, comme le montre l'illustration précédente.

Résumé

Un tableau est un ensemble de données du même type. Les exemples que vous avez étudiés dans cette leçon portent sur des tableaux numériques. Cela étant, n'oubliez pas que les tableaux peuvent aussi contenir n'importe quel type de données, comme des booléens ou des chaînes de caractères.

Vous pouvez créer un tableau dans le diagramme en procédant en deux étapes. Vous placez d'abord une constante de tableau à partir de la palette **Functions»Array** sur le diagramme, puis vous ajoutez la constante ou l'indicateur voulu(e) dans le tableau vierge. Rappelez-vous que vous pouvez aussi créer un tableau sur la face-avant en choisissant les options **Controls»Array & Cluster**, puis en ajoutant la commande ou l'indicateur voulu(e) au tableau vierge.

Les boucles *For* et *While* peuvent accumuler des tableaux à leur bordure. C'est très utile pour créer et manipuler des tableaux.

☞ **Remarque :** ***rappelez-vous que par défaut, LabVIEW active l'indexation sur les boucles For et la désactive sur les boucles While.***

Le polymorphisme est la capacité d'une fonction à s'ajuster à des données d'entrée de types différents. Toutes les fonctions qui acceptent une entrée numérique peuvent ainsi accepter toute représentation numérique, comme un tableau de valeurs numériques ou un *cluster* de valeurs numériques.

Vous pouvez afficher vos données sous la forme de graphes. Ces graphes présentent de nombreuses caractéristiques très utiles que vous pouvez utiliser pour personnaliser vos tracés. Vous pouvez afficher plus d'un tracé à la fois sur un graphe en utilisant la fonction **Build Array** de la palette **Functions»Array**. Le graphe devient alors

automatiquement un graphe multicourbes quand vous câblez le tableau de sortie à son terminal.

De nombreuses fonctions sont à votre disposition pour manipuler les tableaux, comme les fonctions **Build Array** et **Index Array** de la palette **Functions»Array**. Dans les exercices de ce chapitre, vous avez utilisé les fonctions relatives aux tableaux pour travailler uniquement sur des tableaux à une seule dimension. Sachez que ces fonctions sont tout aussi valables pour travailler sur des tableaux à plusieurs dimensions.

Quelques informations supplémentaires

Les tableaux

Nous n'avons abordé dans ce chapitre que quelques-unes des fonctions utiles pour travailler dans les tableaux. LabVIEW en compte beaucoup d'autres dont **Replace Array Element**, **Search 1D Array**, **Sort 1D Array**, **Reverse 1D Array**, **Multiply Array Elements**. Pour en savoir plus sur les tableaux et les fonctions disponibles, veuillez vous reporter au chapitre 15, *Les commandes et les indicateurs de clusters et de tableaux*, du *Manuel de l'utilisateur LabVIEW;* **Online Reference»Function and VI Reference»Array Functions**; et **Online Reference»Function and VI Reference»Cluster Functions**.

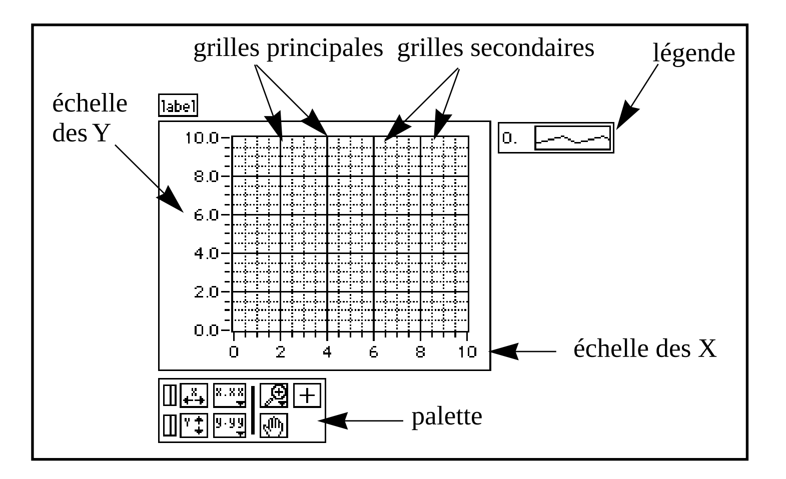
Optimisation de la mémoire et minimisation des copies de données

Pour économiser de la mémoire, vous avez la possibilité d'utiliser des tableaux en simple précision à la place des tableaux en double précision. Afin de mieux comprendre comment LabVIEW exploite la mémoire, veuillez vous reporter à la section *L'utilisation de la mémoire* du chapitre 27, *Performances*, dans le *Manuel de l'utilisateur LabVIEW.*

Personnalisation des graphes

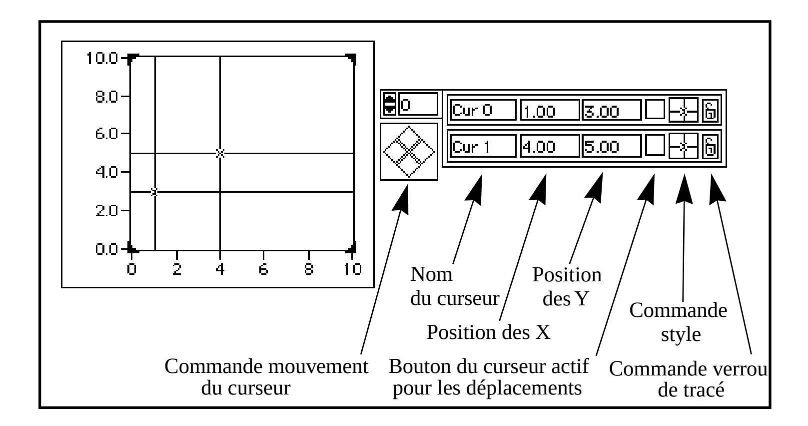
Les graphes oscilloscopiques et les graphes XY comportent un certain nombre d'options que vous pouvez afficher ou au contraire masquer en sélectionnant le sous-menu **Show** du menu local. Parmi ces options, citons la légende qui permet de définir la couleur et le style d'un tracé donné, la palette à partir de laquelle vous pouvez modifier l'échelle et le format des options en cours d'exécution, et pour finir, l'affichage de

curseurs. L'illustration suivante présente toutes ces options, à l'exception de l’affichage du curseur.



Les curseurs des graphes

Dans LabVIEW, vous pouvez placer des curseurs sur tous les graphes et leur attribuer une étiquette. LabVIEW peut programmer le positionnement et la lecture des curseurs. Vous pouvez ainsi verrouiller un curseur sur un tracé ou bien encore déplacer plusieurs curseurs en même temps. Il n'y a pas de limite quant au nombre de curseurs que vous pouvez placer sur un graphe. Dans l'illustration suivante, vous verrez un graphe oscilloscopique qui affiche des curseurs.



Pour plus d'informations sur la personnalisation des graphes, veuillez vous reporter au chapitre 16, *Les commandes et indicateurs de graphes et de graphes déroulants*, du *Manuel de l'utilisateur LabVIEW*.

Consultez le `ZoomGraph.vi` dans `examples\general\graphs\zoom.llb` pour avoir un exemple de lecture de valeurs de curseurs et savoir comment effectuer par programmation des zooms avant et arrière du graphe avec les curseurs.

Les tracés d'intensité

LabVIEW propose deux méthodes pour afficher des données en trois dimensions : le graphe déroulant d'intensité et le graphe d'intensité. Tous deux acceptent des tableaux de nombres à deux dimensions où chaque nombre correspond à une couleur. Vous pouvez définir la couleur désirée de façon interactive en utilisant une échelle de couleurs optionnelle, ou par programmation en utilisant un *attribute node* pour le graphe déroulant. Pour plus d'informations sur les tracés d'intensité, veuillez vous reporter au chapitre 16, *Les commandes et indicateurs de graphes et de graphes déroulants*, du *Manuel de l'utilisateur LabVIEW*. Vous trouverez des exemples de graphes déroulants d'intensité et de graphes d'intensité dans `intgraph.llb` du répertoire `examples\general\graphs`.

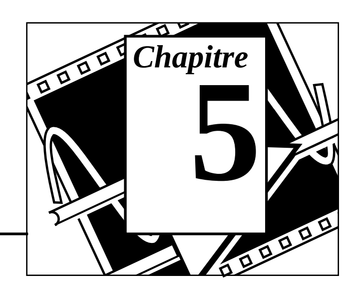
Les tableaux d'acquisition de données (Windows, Macintosh et Sun)

Les données fournies sur une carte d'acquisition enfichable à l'aide des VIs **Data Acquisition** peuvent se présenter sous forme de valeurs individuelles, de tableaux à une dimension ou encore de tableaux à deux dimensions. Veuillez vous reporter à la section intitulée *Data Organization in for Analog Applications* du chapitre 3, *Basic Data Acquisition Concepts*, dans *LabVIEW Data Acquisition Basics Manual* pour en savoir davantage sur la manipulation des tableaux de données.

Exemples de graphes

Vous trouverez plusieurs exemples de graphes dans `examples\general\graphs`. Il contient des VIs permettant de mettre en œuvre plusieurs fonctions dans des tableaux et des graphes.

Structures Condition, structures Séquence et boîte de calcul



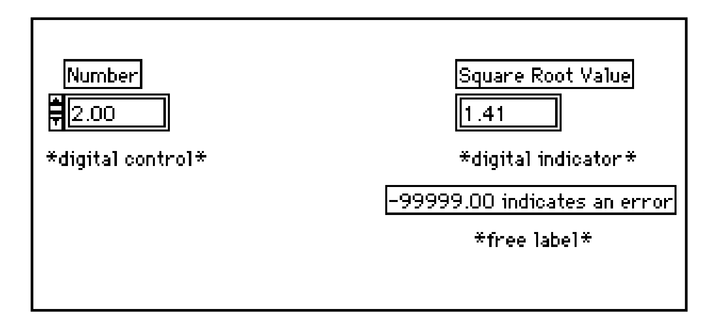
Vous allez apprendre :

- Comment utiliser la structure Condition.
- Comment utiliser la structure Séquence.
- Ce que sont les variables locales de séquence et comment les utiliser.
- Ce qu'est une boîte de calcul (*formula node*) et comment l'utiliser.
- Pour consulter des exemples de ces structures, reportez-vous au répertoire `examples\general\structs.llb`.

Mise en œuvre d'une structure Condition

OBJECTIF Construire un VI qui teste un nombre pour voir s'il est positif ou non. S'il est positif, le VI calcule alors sa racine carrée, dans le cas contraire, il signale une erreur.

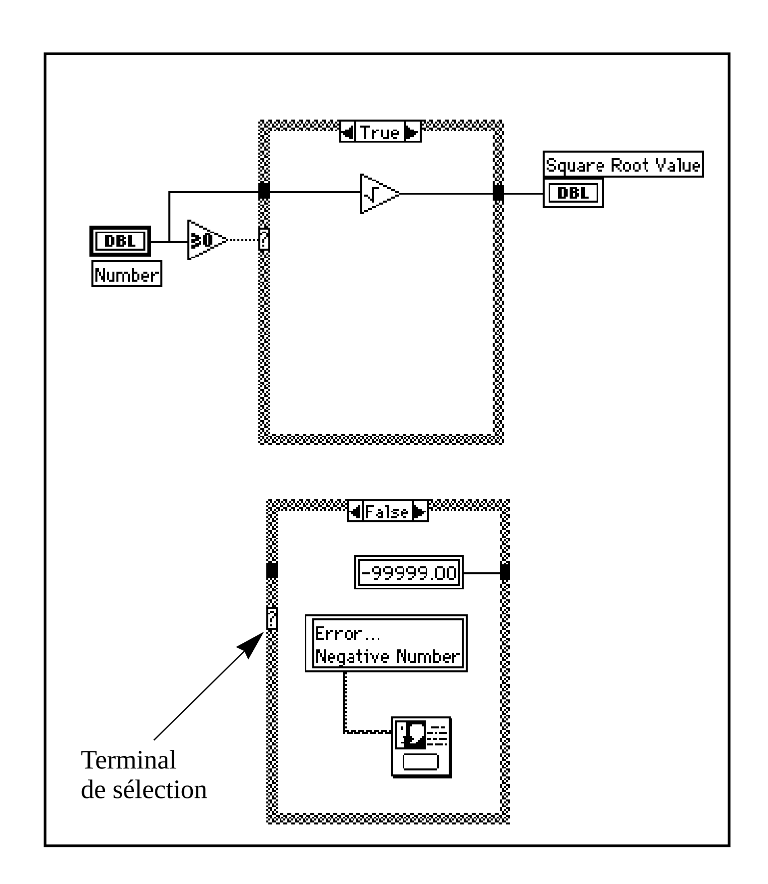
La face-avant



1. Ouvrez une nouvelle fenêtre de face-avant et construisez la face-avant de l'illustration précédente.

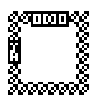
La commande **Number** fournit le nombre. L’indicateur **Square Root Value** affiche la racine carrée de ce nombre. Le texte libre fait office de remarque à l’attention de l’utilisateur.

Le diagramme





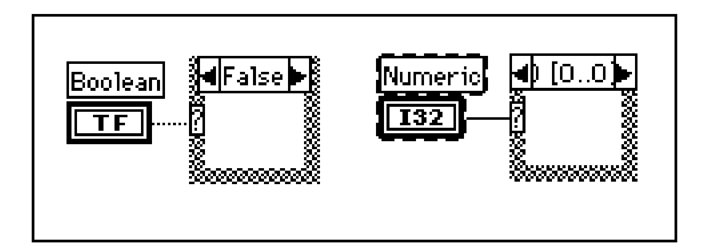
1. Ouvrez la fenêtre du diagramme.



2. Placez une structure Condition (**Functions»Structures**) dans le diagramme. Agrandissez la structure Condition en tirant sur l’un de ses coins à l’aide du curseur de redimensionnement.

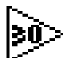
Par défaut, la structure Condition est booléenne et ne peut présenter que deux conditions, à savoir : TRUE ou FALSE. Une structure Condition booléenne s’apparente aux instructions if-then-else des langages de programmation à base de texte. Elle passe

automatiquement en numérique lorsque vous câblez une commande numérique sur le terminal de sélection.

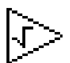


Vous ne pouvez afficher qu'une seule condition à la fois. Pour passer d'une condition à l'autre, il suffit de cliquer sur les flèches qui figurent en haut de la structure Condition.

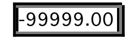
3. Choisissez les autres objets présents dans le diagramme et câblez-les en vous inspirant du diagramme de l'illustration précédente.



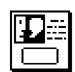
Fonction **Greater Or Equal To 0?** (**Functions»Comparison**). Dans cet exercice, cette fonction va nous permettre de déterminer si l'entrée est négative. La fonction retourne l'état TRUE si le nombre en entrée est supérieur ou égal à 0.



Fonction **Square Root** (**Functions»Numeric**). Dans cet exercice, cette fonction calcule la racine carrée du nombre en entrée.



Numeric constant (**Functions»Numeric**).



Fonction **One Button Dialog** (**Functions»Time & Dialog**). Dans cet exercice, cette fonction affiche une boîte de dialogue qui contient le message `Error...Negative Number`.



String Constant (**Functions»String**). Vous entrez du texte à l'intérieur de la boîte avec l'outil Texte.

Dans cet exercice, l'exécution du VI conduit à deux choix possibles : TRUE ou FALSE. Si le nombre est supérieur ou égal à zéro, le VI est alors dans la situation TRUE et calcule la racine carrée du nombre. Dans la situation FALSE, la sortie indique `-99999.00` et la boîte de dialogue affiche le message suivant : `Error...Negative Number`.

☞ **Remarque :** ***vous devez définir le tunnel de sortie pour chaque cas. Lorsque vous créez un tunnel de sortie pour un cas particulier, vous créez automatiquement un tunnel au même endroit dans les autres cas. Lorsqu'un tunnel n'est pas câblé, il prend la forme d'un carré blanc.***

Veillez à câbler le tunnel de sortie sur chaque cas en cliquant sur le tunnel de sortie lui-même à chaque fois. Dans cet exercice, vous allez affecter une valeur au tunnel de sortie du cas FALSE puisque le cas TRUE a déjà un tunnel de sortie. Si vous ne souhaitez pas affecter une valeur de sortie à tous les cas, vous devez placer l'indicateur dans ce cas précis ou bien encore utiliser une variable locale ou une variable globale.

4. Revenez sur la face-avant et lancez le VI. Prenez un nombre supérieur à zéro et un nombre inférieur à zéro en modifiant la valeur de la commande numérique que vous avez nommée `Number`. Vous remarquerez que lorsque la commande numérique prend une valeur négative, LabVIEW affiche le message d'erreur que vous avez attribué au cas FALSE de la structure Condition.
5. Enregistrez et fermer le VI. Nommez-le `My Square Root.vi`.

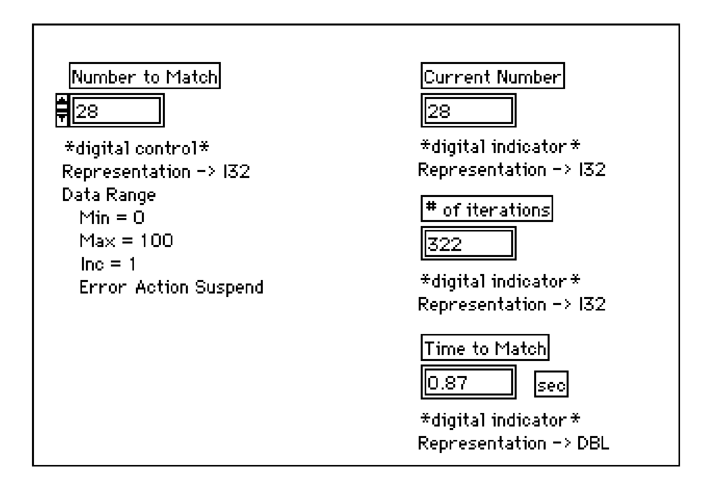
Logique du VI

```
if (Number >= 0) then
Square Root Value = SQRT(Number)
else
Square Root Value = -99999.00
Display Message ”Error...Negative Number”
end if
```

Mise en œuvre d'une structure Séquence

OBJECTIF Construire un VI qui calcule le temps nécessaire pour générer un nombre aléatoire qui corresponde à un nombre donné.

La face-avant



1. Ouvrez et construisez une nouvelle face-avant en vous inspirant de l'illustration suivante. Veillez à bien respecter les consignes données suivantes pour modifier les commandes et les indicateurs.

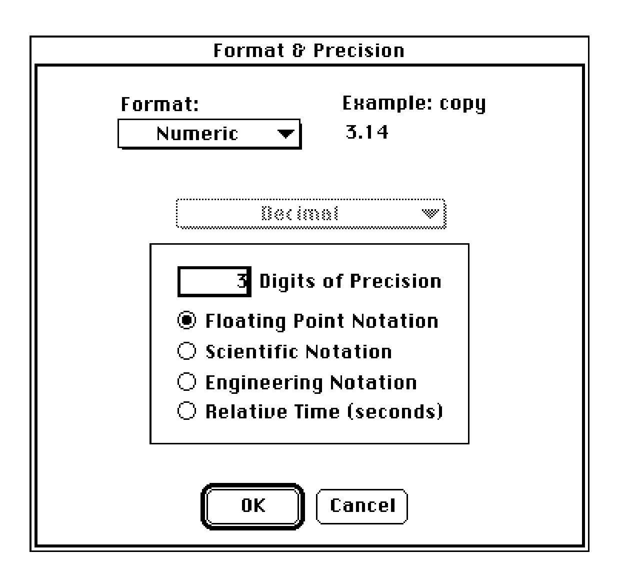
La commande `Number to Match` contient le nombre que vous souhaitez atteindre. L'indicateur `Current Number` affiche la valeur aléatoire générée. L'indicateur # `of iterations` fournit le nombre d'itérations nécessaires à la recherche. Enfin, l'indicateur `Time to Match` indique combien de secondes il a fallu pour obtenir le nombre correspondant.

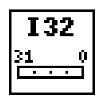
Modification du format numérique

Par défaut, LabVIEW affiche les valeurs des commandes numériques au format décimal, avec deux chiffres après la virgule (symbolisée ici par un point), comme par exemple 3.14. Vous pouvez aussi utiliser l'option **Format & Precision...** d'une commande ou d'un indicateur d'un menu local pour modifier la précision ou afficher les commandes et les indicateurs numériques conformément aux usages des

scientifiques et des ingénieurs. Vous pouvez aussi utiliser l'option **Format & Precision...** pour horodater les valeurs numériques.

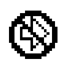
1. Modifiez la précision de l'indicateur `Time to Match`.
   a. Accédez au menu local de l'indicateur numérique `Time to Match` puis choisissez **Format & Precision...**. Vous devez être dans la face-avant pour pouvoir accéder à ce menu.
   b. Entrez la valeur `3` dans le paramètre `Digits of Precision`, puis cliquez sur **OK**.





2. Modifiez la représentation de la commande numérique et transformez deux des indicateurs numériques en entiers longs.
   a. Ouvrez un menu local sur la commande numérique `Number to Match`, puis choisissez **Representation»Long**.
   b. Répétez l'étape précédente pour les indicateurs numériques `Current Number` et `# of iterations`.

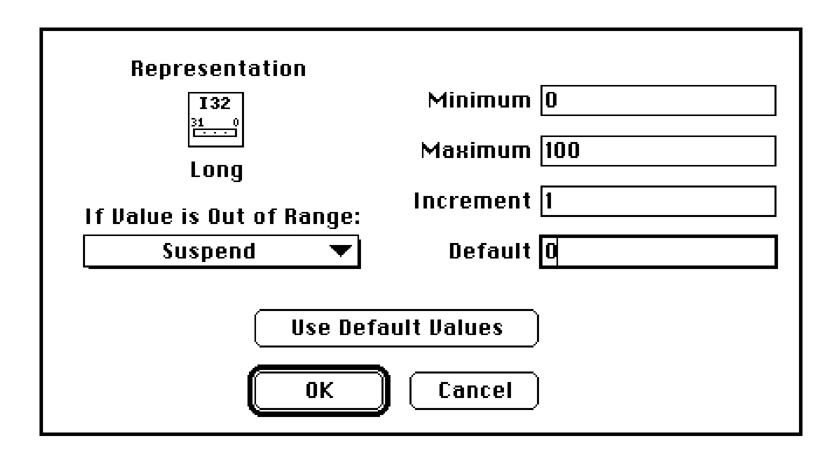
Délimitation de la gamme de données



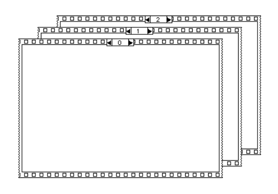
Avec l'option **Data Range...**, vous pouvez éviter qu'un opérateur ne positionne la valeur d'une commande ou d'un indicateur en dehors d'une gamme ou d'une incrémentation prédéfinie. Vous pouvez soit ignorer la valeur, la forcer à rester à l'intérieur de la gamme prédéfinie ou bien encore interrompre définitivement l'exécution. Le symbole d'erreur de gamme remplace le bouton Exécution dans la barre d'outils, lorsqu'une erreur de gamme interrompt l'exécution. De plus, un cadre sombre et épais entoure la commande incriminée.



1. Fixez la gamme entre 0 et 100 avec un incrément de 1.
   a. Ouvrez un menu local sur l'indicateur `Time to Match` puis choisissez **Data Range...**.
   b. Complétez la boîte de dialogue comme indiqué dans l'illustration suivante, puis cliquez sur **OK**.



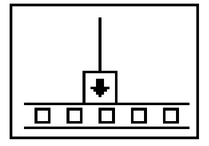
Le diagramme



1. Ouvrez la fenêtre du diagramme.
2. Placez la structure Séquence (**Functions»Structures**) dans la fenêtre du diagramme.

La structure Séquence qui se présente un peu comme les trames successives d’un film, exécute séquentiellement des diagrammes. Dans les langages de programmation classiques, les instructions s’exécutent suivant leur ordre d’apparition. Alors que dans la programmation par flux des données, un nœud ne s’exécute que lorsque toutes les données sont disponibles à ses entrées. Toutefois, il est parfois nécessaire d’exécuter un nœud avant un autre. Le cas échéant, LabVIEW utilise la structure Séquence comme une méthode permettant de contrôler l’ordre d’exécution des nœuds. LabVIEW place le diagramme que le VI exécute en premier à l’intérieur du cadre 0, puis place le diagramme qu’il exécute en deuxième à l’intérieur du cadre 1, et ainsi de suite. Comme dans le cas de la structure Condition, vous ne pouvez voir qu’un seul cadre à la fois.

3. Agrandissez la structure en tirant sur l’un de ses coins à l’aide du curseur de redimensionnement.
4. Pour créer un nouveau cadre, ouvrez un menu local en bordure du cadre et choisissez **Add Frame After**. Procédez de même pour créer le cadre 2.

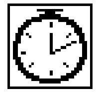


Le cadre 0 de l’illustration précédente contient un petit carré avec une flèche à l’intérieur. Ce petit carré représente une variable locale qui achemine les données entre les cadres d’une structure Séquence. Vous pouvez créer des variables locales sur la bordure des cadres. Les données câblées à la variable locale d’un cadre sont alors disponibles dans les cadres suivants. Néanmoins, il convient de noter que vous ne pouvez pas accéder aux données des cadres précédant la création du cadre contenant la variable locale.

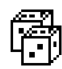
5. Créez cette variable locale en ouvrant un menu local sur la bordure inférieure du cadre 0 puis en choisissant **Add Sequence Local**.

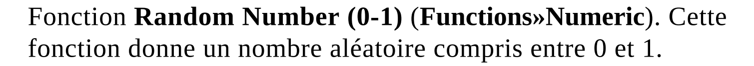
La variable locale se présente sous la forme d’un carré vide. La flèche à l’intérieur du carré apparaît automatiquement dès que vous câblez une fonction à la variable locale.

6. Terminez le diagramme jusqu’à ce qu’il ressemble à celui de l’illustration reproduite dans la section *Diagramme*.

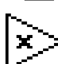


Fonction **Tick Count (ms)** (**Functions»Time & Dialog**). Cette fonction donne le nombre de millisecondes qui se sont écoulées depuis la mise sous tension. Pour cet exercice, vous aurez besoin de deux fonctions **Tick Count**.





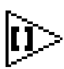
Fonction **Random Number (0-1)** (**Functions»Numeric**). Cette fonction donne un nombre aléatoire compris entre 0 et 1.



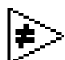
Fonction **Multiply** (**Functions»Numeric**). Dans cet exercice, cette fonction multiplie le nombre aléatoire par 100. C'est-à-dire qu'elle génère un nombre aléatoire compris entre 0.0 et 100.0.



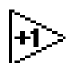
Fonction **Numeric Constant** (**Functions»Numeric**). Dans cet exercice, la constante numérique représente le nombre maximum pouvant être multiplié.



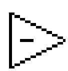
Fonction **Round to Nearest** (**Functions»Numeric**). Dans cet exercice, cette fonction arrondit le nombre aléatoire compris entre 0 et 100 au nombre entier le plus proche.



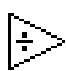
Fonction **Not Equal?** (**Functions»Comparison**). Dans cet exercice, cette fonction compare le nombre aléatoire au nombre indiqué dans la face-avant et donne l'état TRUE si les deux nombres sont différents. Dans le cas contraire, cette fonction donne l'état FALSE.



Fonction **Increment** (**Functions»Numeric**). Dans cet exercice, cette fonction incrémente le comptage de la boucle *While* de 1.



Fonction **Subtract** (**Functions»Numeric**). Dans cet exercice, cette fonction fournit le temps écoulé entre le cadre 2 et le cadre 0 en millisecondes.



Fonction **Divide** (**Functions»Numeric**). Dans cet exercice, cette fonction divise le nombre de millisecondes écoulées par 1000 pour convertir le nombre en secondes.



Fonction **Numeric Constant** (**Functions»Numeric**). Dans cette exercice, cette fonction convertit le nombre exprimé en millisecondes en secondes.

Dans le cadre 0, la fonction **Tick Count (ms)** fournit le temps en millisecondes. Cette valeur est câblée à la variable locale, pour être transmise aux séquences suivantes. Dans le cadre 1, le VI exécute la boucle *While* tant que le nombre indiqué ne correspond pas au nombre délivré par la fonction **Random Number (0-1)**. Dans le cadre 2, la fonction **Tick Count (ms)** donne le nouveau temps exprimé en millisecondes. Le VI soustrait le temps enregistré dans la séquence 0 (transmis par la variable locale) au nouveau temps, de façon à calculer le temps écoulé lors de la séquence 1.

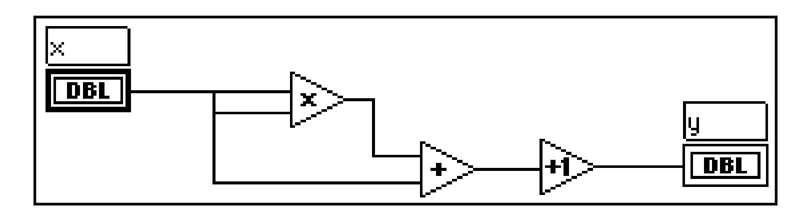
7. Revenez dans la face-avant puis entrez un nombre à l'intérieur de la commande `Number to Match` puis lancez le VI.
8. Enregistrez et fermez le VI. Nommez-le `My Time to Match.vi`.

La boîte de calcul

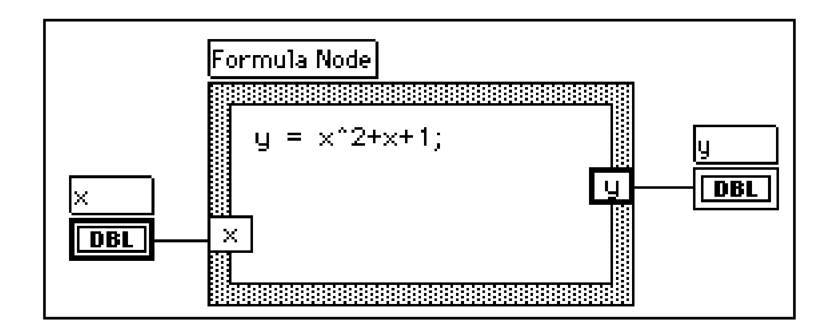
La boîte de calcul *(formula node)* est une fenêtre réajustable qui permet de saisir des formules mathématiques directement dans un diagramme. Pour ce faire, vous placez la boîte de calcul dans le diagramme à partir de la palette **Function»Structures**. Cette méthode est très pratique lorsque l'équation contient plusieurs variables ou qu'elle est relativement compliquée. Par exemple, considérez l'équation suivante :

$y = x^2 + x + 1$

Si vous effectuez cette équation en utilisant les fonctions arithmétiques habituelles de LabVIEW, vous obtiendrez un diagramme semblable à celui reproduit ci-dessous.



Vous pouvez effectuer cette même équation en utilisant une boîte de calcul, identique à celle de l'illustration suivante.



Grâce à la boîte de calcul, vous pouvez entrer directement une formule complexe au lieu de créer plusieurs sous-sections de diagrammes. Vous entrez les formules avec l'outil Texte. Vous créez les terminaux d'entrée et de sortie de la boîte de calcul en ouvrant un menu local en bordure du nœud et en choisissant **Add Input** (**Add Output**). Tapez ensuite le nom de la variable dans le carré. Sachez que les variables

sont différentes selon que vous utilisez des majuscules ou des minuscules pour les saisir. Vous entrez la ou les formule(s) mathématique(s) à l'intérieur du carré. Chaque élément de la formule mathématique doit impérativement se terminer par un point-virgule (;).

La liste des opérateurs et des fonctions disponibles dans la boîte de calcul figure dans la fenêtre d'aide correspondante, comme le montre l'illustration suivante. Un point-virgule se trouve à la fin de chaque formule. Les fonctions relatives à la boîte de calcul (**Formula Node**) sont reprises en détail au chapitre 20, *La boîte de calcul*, du *Manuel de l'utilisateur LabVIEW*.

```
Formula Node operators, lowest precedence first:
=                        assignment
? :                      conditional
||  &&                   logical
==  !=  >  <  >=  <=     relational
+  -  *  /  ^            arithmetic
+  -  !                  unary

Formula Node functions:
abs  acos  acosh  asin  asinh  atan  atanh  ceil
cos  cosh  cot  csc  exp  expm1  floor  getexp  getman
int  intrz  ln  lnp1  log  log2  max  min  mod  rand
rem  sec  sign  sin  sinc  sinh  sqrt  tan  tanh
```

L'exemple suivant vous explique comment réaliser un branchement conditionnel à l'intérieur d'une boîte de calcul.

Imaginez que vous vouliez calculer la racine carrée de *x*. Si *x* est positif, le résultat du calcul est attribué à *y*. Si *x* est négatif, le code -99 est attribué à *y*.

```
if (x >= 0) then

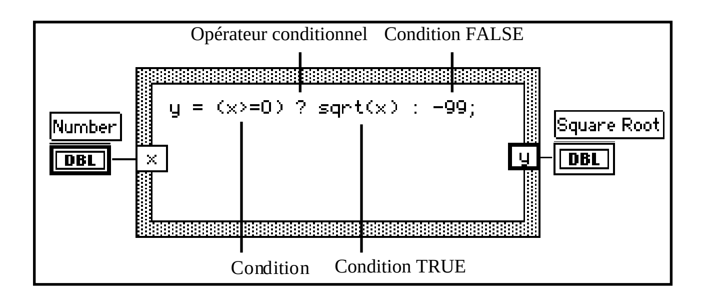
y = sqrt(x)

else

y = -99

end if
```

Vous pouvez parvenir au même résultat en utilisant une boîte de calcul, comme celle reproduite dans le diagramme ci-après.



Mise en œuvre d’une boîte de calcul

OBJECTIF Construire un VI qui utilise un *Formula Node* pour calculer les équations suivantes :

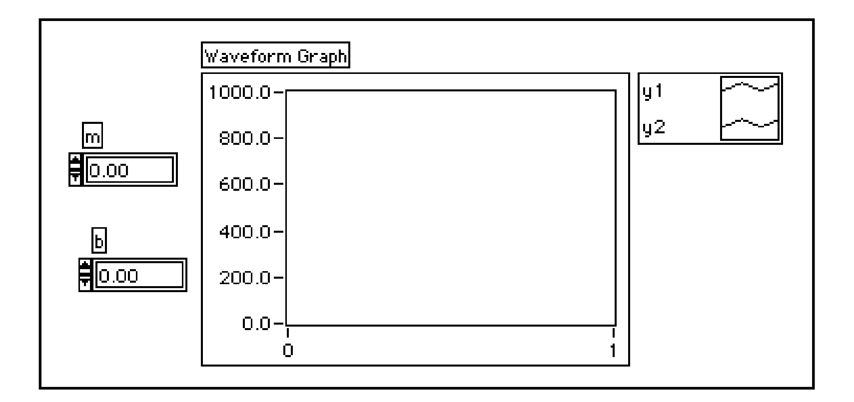
$y1 = x^3 - x^2 + 5$

y2 = m * x + b

où *x* varie entre 0 et 10.

Vous n’utiliserez qu’une seule boîte de calcul pour résoudre les deux équations en superposant les résultats dans un seul graphe.

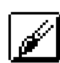
La face-avant



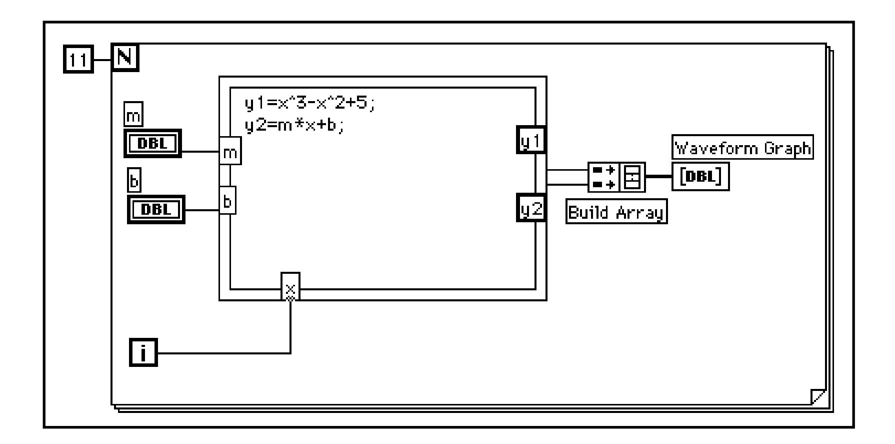
1. Ouvrez une nouvelle fenêtre de face-avant puis construisez la face-avant présentée dans l'illustration précédente. L'indicateur du graphe oscilloscopique affiche les tracés de l'équation. Le VI utilise les deux commandes numériques pour entrer les valeurs à attribuer à *m* et à *b*.

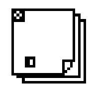


Composez la légende du graphe de l'illustration précédente en choisissant **Show»Legend**. Utilisez le curseur de redimensionnement pour faire descendre la légende afin d'afficher les deux tracés. Utilisez l'outil Texte pour renommer les tracés. Vous avez la possibilité de définir le style de ligne pour chacun des tracés en cliquant sur la légende pour ouvrir un menu local. Vous pouvez également affecter une couleur aux tracés à l'aide de l'outil Pinceau, en cliquant sur les tracés dans la légende.

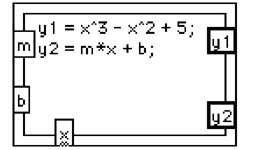


Le diagramme



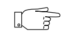


1. Reproduisez le diagramme de l'illustration précédente.
2. Placez la boucle *For* (**Functions»Structures**) dans le diagramme et tirez sur le coin pour agrandir la boucle.

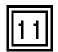


Formula Node (**Functions»Structures**). Avec cette boîte de calcul, vous pouvez entrer directement une ou plusieurs formule(s). Créez trois terminaux d'entrée en ouvrant un menu local en bordure du cadre et en choisissant **Add Input**. De même, vous créez le terminal de sortie en choisissant **Add Output** à partir du menu local.

Lorsque vous créez un terminal d'entrée ou de sortie, vous devez lui donner un nom de variable. Ce nom doit exactement correspondre au nom que vous utilisez dans la formule. Des noms apparemment identiques seront pourtant différents s'ils sont en majuscules ou en minuscules. Par exemple, si vous utiliser un a minuscule pour nommer le terminal, vous devez obligatoirement utiliser ce même a minuscule dans la formule. Pour cela, vous pouvez entrer les noms de variables et de formules avec l'outil Texte.



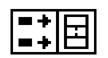
Remarque : ***même si les noms de variables ne sont pas limités en longueur, ne perdez pas de vue que des noms trop longs vont occuper beaucoup d'espace dans le diagramme. Un point-virgule (;) doit être placé à la fin de chaque formule.***



Numeric Constant (Functions»Numeric). Vous pouvez également ouvrir un menu local sur le terminal de comptage et choisir **Create Constant** pour créer et câbler automatiquement la constante numérique. Celle-ci indique le nombre d'itérations de la boucle *For*. Si x varie entre 0 et 10 inclus, vous devez câbler 11 au terminal de comptage.



Comme le terminal d'itération compte de 0 à 10, il sert à contrôler la valeur *X* dans la boîte de calcul.



Build Array (**Functions»Array**) rassemble les deux entrées du tableau en un graphe multicourbes. Créez les deux terminaux d'entrée en utilisant le curseur de redimensionnement pour tirer sur l'un des coins.

3. Revenez dans la face-avant et lancez le VI avec différentes valeurs pour *m* et *b*.
4. Enregistrez et fermez le VI. Nommez-le `My Equations.vi`.

Résumé

LabVIEW est doté de deux structures qui permettent de contrôler le flux des données. Il s'agit des structures Condition et Séquence. LabVIEW identifie ces deux structures à une pile de cartes : vous ne voyez qu'un seul cadre à la fois.

La structure Condition permet de brancher différents sous-diagrammes en fonction de l'entrée du terminal de sélection. Vous placez les sous-diagrammes à l'intérieur du cadre de chaque condition de la structure Condition. La valeur transmise au terminal de sélection peut être booléenne (2 conditions) ou numérique (jusqu'à $2^{31} - 1$). LabVIEW détermine automatiquement le type de terminal de sélection lorsque vous câblez une commande booléenne ou numérique.

La structure Séquence permet d'exécuter un diagramme selon un ordre particulier. Vous placez la partie du diagramme que vous souhaitez exécuter en premier dans le cadre 0 de la structure Séquence, puis le diagramme que vous souhaitez exécuter en deuxième dans le cadre 1, et ainsi de suite.

Les variables locales de séquence permettent d'acheminer des valeurs entre les cadres de la structure Séquence. Les données transférées via une variable locale ne sont disponibles que dans les cadres suivants et non dans les cadres précédents.

La boîte de calcul permet d'entrer directement des formules mathématiques dans un diagramme. Cette possibilité est très utile dès lors que l'équation d'une fonction comporte plusieurs variables ou lorsqu'elle est complexe. N'oubliez pas que les noms de variables n'ont pas la même signification selon que vous les écrivez en majuscules ou en minuscules et que vous devez obligatoirement placer un point-virgule (;) à la fin de chaque formule.

Quelques informations supplémentaires

Les structures Condition et Séquence

Pour plus d'informations sur les structures Condition et Séquence, veuillez vous reporter à la section *Les structures Condition et Séquence* du chapitre 19, *Structures,* du *Manuel de l'utilisateur LabVIEW*. Ces structures étant par ailleurs des structures fondamentales dans le concept de programmation de LabVIEW, vous pouvez observer leur utilisation dans la plupart des VIs du répertoire `examples`.

Le temps de cycle des structures Séquence

Les structures Séquence servent souvent à calculer le temps d'exécution d'une fonction ou d'un VI. L'exemple `Timing Template.vi` dans `examples\general\structs.llb` indique la procédure à suivre pour cette opération.

Les boîtes de calcul

Vous trouverez des informations supplémentaires sur les boîtes de calcul au chapitre 20, *La boîte calcul,* du *Manuel de l'utilisateur LabVIEW*.

La dépendance artificielle des données

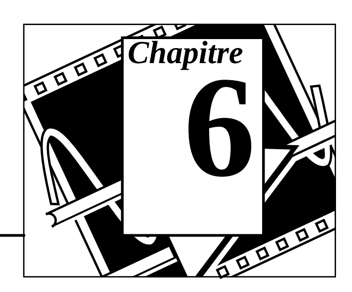
Les nœuds qui ne sont pas connectés entre eux peuvent s'exécuter dans n'importe quel ordre et non pas obligatoirement de gauche à droite ou de haut en bas. La structure Séquence constitue l'une des méthodes

pour contrôler l'ordre d'exécution lorsqu'il n'existe pas de dépendance naturelle entre les données.

Une autre méthode pour contrôler l'ordre d'exécution consiste à créer une *dépendance artificielle des données*, un principe selon lequel, c'est l'ordre *d'arrivée* des données, et non leur valeur, qui déclenche l'exécution d'un objet. En fait, le récepteur peut ne pas utiliser les données en interne. L'intérêt de la dépendance artificielle réside dans le fait que tous les nœuds sont visibles sur un seul niveau, même s'il arrive parfois que les liens artificiels entre les nœuds soient source de confusion.

Vous pouvez ouvrir `Timing Template (data dep).vi` dans `examples\general\structs.llb` pour voir comment le `Timing Template` a été modifié en utilisant la dépendance artificielle des données plutôt que la structure Séquence.

Chaînes de caractères et E/S sur fichier



Vous allez apprendre :

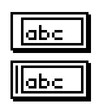
- Comment créer des indicateurs et des commandes de type chaînes de caractères.
- Comment utiliser les fonctions de manipulation de chaînes de caractères.
- A manipuler les entrées et sorties sur fichier.
- Comment enregistrer des données dans un fichier au format tableur.
- Comment écrire ou lire des données de fichiers texte.

Les chaînes de caractères

Une chaîne de caractères est un ensemble de caractères ASCII. Leur utilisation va bien au-delà du cadre des messages textuels. Dans le domaine du contrôle d'instruments, vous pouvez transmettre des données numériques sous la forme de chaînes de caractères, puis convertir ces chaînes en chiffres. Le stockage de données numériques sur disque peut se faire sous la forme de chaînes de caractères. Pour stocker des nombres dans un fichier ASCII, il vous faut d'abord convertir ces nombres en chaînes de caractères avant de les écrire dans un fichier sur le disque.

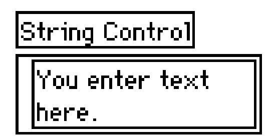
Pour consulter des exemples de chaînes de caractères, reportez-vous au répertoire `examples\general\strings.llb`.

Création des indicateurs et commandes de type chaînes de caractères



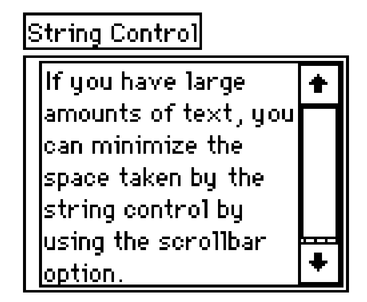
Vous retrouverez les indicateurs et commandes de type chaînes de caractères présentés à gauche dans la palette **Controls»String & Table**. Vous pouvez entrer ou bien encore modifier du texte à l'intérieur d'une commande de chaînes de caractères avec l'outil Doigt ou l'outil Texte.

Pour agrandir les indicateurs et commandes, cliquez sur un coin et faites-le glisser à l'aide de l'outil Doigt.



Les chaînes de caractères et E/S sur fichier

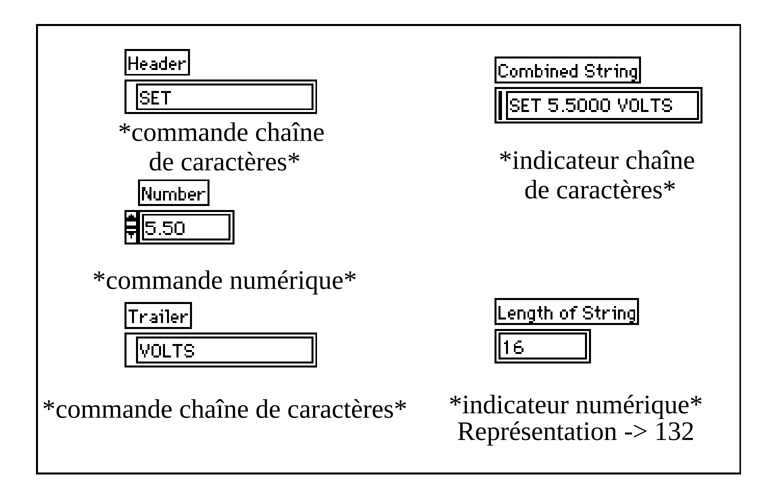
Si vous souhaitez réduire au minimum l’espace occupé par une commande ou un indicateur de chaînes sur la face-avant, choisissez l’option **Show»Scrollbar**. Si cette option est grisée, il vous faut agrandir la fenêtre pour la rendre visible.



Mise en œuvre des fonctions de chaînage

OBJECTIF LabVIEW offre de nombreuses fonctions permettant de manipuler les Fchaînes de caractères. Ces fonctions sont disponibles dans la palette **Functions»String**. Vous allez construire un VI qui convertit un nombre en chaîne de caractères et qui va ensuite concaténer cette chaîne à d’autres chaînes de caractères pour ne donner qu’une seule chaîne de caractères. Le VI détermine également la longueur de la chaîne obtenue.

La face-avant

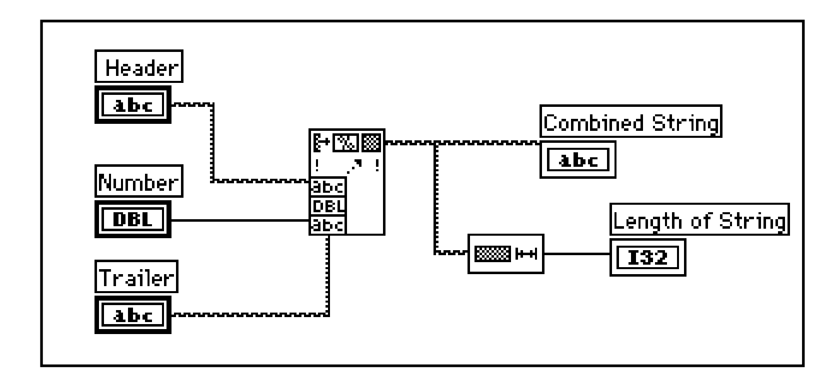


1. Ouvrez et construisez une nouvelle face-avant en vous inspirant de celle de l'illustration précédente. Veillez à modifier correctement les indicateurs et les commandes comme indiqué.

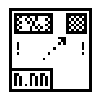
Les deux commandes de chaînes de caractères et la commande numérique peuvent parfaitement se combiner pour ne former qu'une seule chaîne de caractères de sortie, et s'afficher dans l'indicateur de chaîne de caractères. L'indicateur numérique sert à afficher la longueur de la chaîne.

Dans cet exercice, la sortie **Combined String** présente un format comparable aux chaînes des commandes utilisées pour communiquer avec les instruments GPIB (IEEE 488) et série (RS-232 ou RS-422). Pour en savoir plus sur les chaînes utilisées dans les commandes d'instruments, veuillez vous reporter au chapitre 8, intitulé *L'acquisition de données et le contrôle d'instruments*, de ce même tutorial.

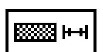
Le diagramme



1. Construisez le diagramme de l'illustration précédente.



La fonction **Format Into String** (**Functions»String**) permet de concaténer et de formater les nombres et les chaînes pour ne former qu'une seule chaîne de sortie. Utilisez le curseur de redimensionnement et placez-le sur l'icône pour ajouter trois entrées supplémentaires.



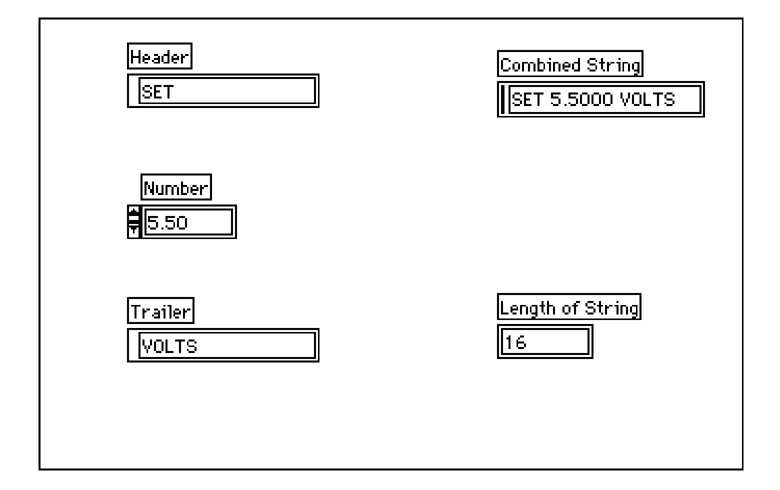
La fonction **String Length** (**Functions»String**) indique le nombre de caractères contenus dans la chaîne concaténée.

2. Lancez le VI. Vous remarquerez que la fonction **Format Into String** a concaténé les deux commandes de chaînes de caractères et la commande numérique en une seule chaîne de caractères en sortie.
3. Enregistrez le VI sous le nom `My Build String.vi`. Vous reprendrez ce VI au cours du prochain exercice.

Mise en œuvre des chaînes de formatage

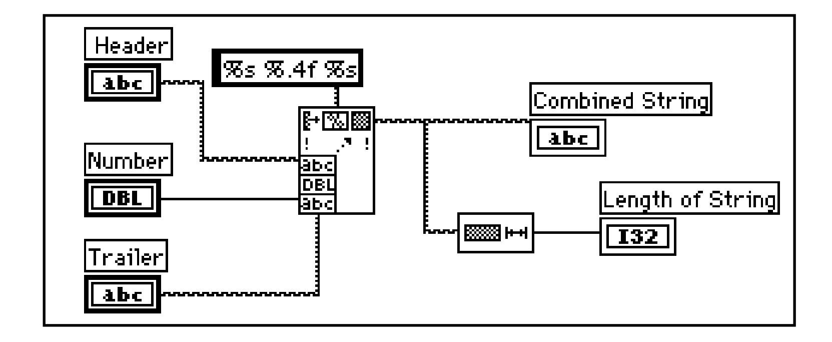
OBJECTIF Utiliser le VI **My Build String** de l'exercice précédent pour créer une chaîne de caractères de format. Cette opération va vous permettre d'indiquer le format des arguments, y compris la longueur du champ, la base (hexadécimale, octale, etc.), ainsi que le texte servant à séparer les arguments.

La face-avant

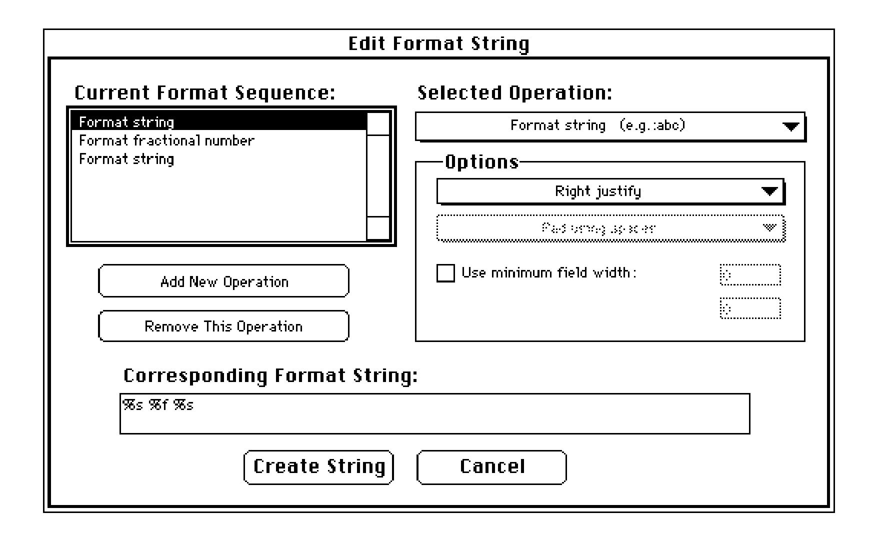


1. Ouvrez le VI **My Build String** que vous avez créé dans l'exercice précédent.

Le diagramme



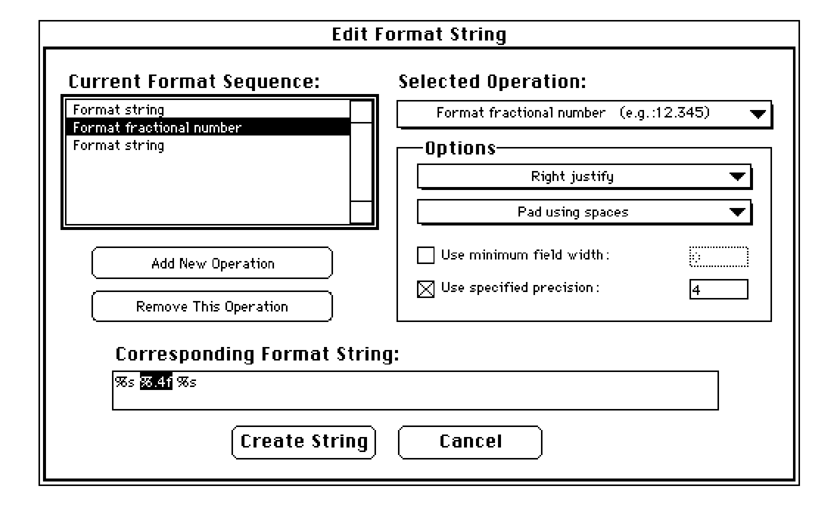
1. Ouvrez un menu local sur **Format Into String** puis choisissez l’option **Edit Format String**. Vous obtenez ainsi la boîte de dialogue suivante.



☞ **Remarque :** ***vous pouvez également double-cliquer sur le nœud pour accéder directement à la boîte de dialogue* Edit Format String.**

Vous remarquerez que **Current Format Sequence** contient les types d’argument, dans l’ordre où vous les avez câblés.

2. Réglez la précision des valeurs numériques sur `4`.
   a. Mettez en surbrillance l'option `Format fractional number` qui apparaît dans la liste de la boîte de dialogue **Current Format Sequence**.
   b. Cliquez sur la case à cocher **Use Specified Precision**.
   c. Mettez en surbrillance la valeur numérique en regard de la case libellée **Use Specified Precision**, tapez le chiffre `4`, puis appuyez sur la touche <`Enter`> (Windows) <`Return`> (Macintosh) <`Return`> (Sun) ou <`Enter`> (HP-UX). L'illustration suivante présente les options choisies pour régler la précision d'un nombre.

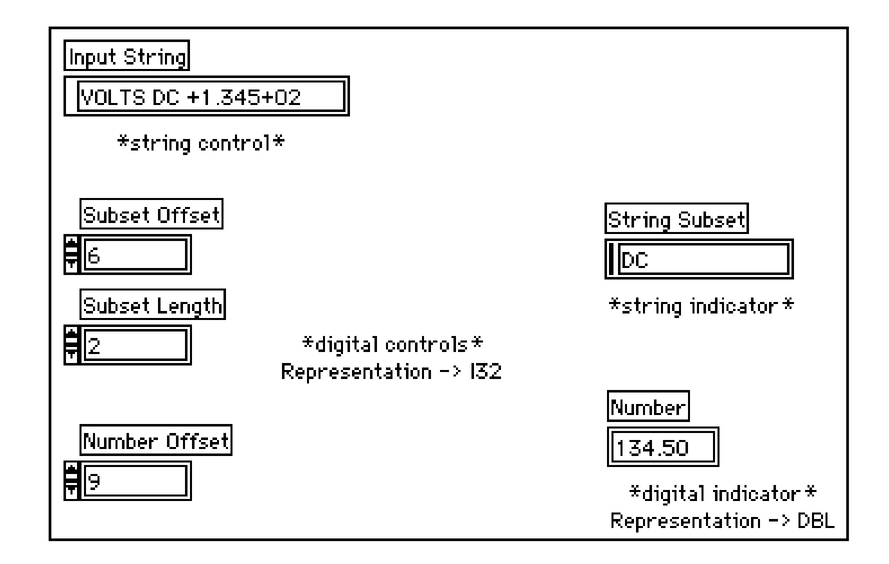


3. Appuyez sur le bouton **Create String**. Le seul fait d'appuyer sur ce bouton insère automatiquement les informations nécessaires pour obtenir le format qui convient et câble le format de la chaîne à la fonction.
4. Revenez à la face-avant et tapez du texte à l'intérieur des deux commandes chaînes de caractères ainsi qu'un nombre à l'intérieur de la commande numérique. Exécutez le VI.
5. Enregistrez le VI sous le nom `My Format String.vi`.

Autres fonctions de chaînage

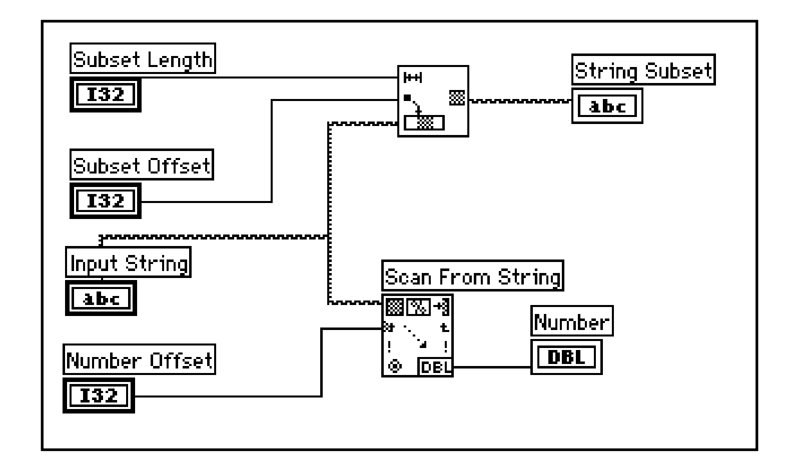
OBJECTIF Dans l'exercice précédent, vous avez utilisé les fonctions de chaînage pour créer une longue chaîne à partir de nombres et de chaînes de plus petite taille. Dans l'exercice suivant, vous allez étudier un VI qui extrait les informations d'une chaîne de caractères plus longue. Vous allez prendre un sous-ensemble d'une chaîne qui représente un nombre et le convertir en une valeur numérique.

La face-avant



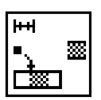
1. Ouvrez `Parse String.vi` dans `examples\general\strings.llb.` Exécutez le VI avec les entrées par défaut. Vous remarquerez que le sous-ensemble *DC* est extrait de la chaîne d'entrée. Par ailleurs, la partie numérique de la chaîne, une fois extraite, a été convertie en un nombre. Vous pouvez essayer différentes valeurs de commande. N'oubliez pas que les chaînes, à l'instar des tableaux, sont indexées à partir de zéro. Vous pouvez aussi afficher le diagramme pour voir comment s'effectue l'extraction des éléments de la chaîne d'entrée.

Le diagramme



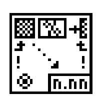
1. Ouvrez le diagramme du VI **Parse String**, représenté ci-dessus.

LabVIEW a recours aux fonctions **String Subset** et **From Exponential/Fract/Eng** pour effectuer la déconcaténation de la chaîne d'entrée.



La fonction **String Subset** (**Functions»String)** génère la sous-chaîne commençant par **offset** et par le nombre de caractères **length**. Le premier caractère offset est égal à zéro.

En maintes occasions, vous aurez à convertir des chaînes de caractères en nombres, comme par exemple lorsque vous convertissez une chaîne de données envoyée par un instrument en valeurs de données.



La fonction **Scan From String** (**Functions»String**) analyse une chaîne de caractères et convertit les caractères numériques valides (de 0 à 9, +, -, e, E, et le point) en nombres. Si vous câblez une chaîne formatée, la fonction **Scan From String** effectue les conversions en fonction du format retenu. Sinon, la fonction **Scan From String** effectue des conversions par défaut pour chaque terminal d'entrée correspondant à la fonction. Cette fonction analyse la chaîne (**string**) à partir d'**offset**. La position **offset** du premier caractère est égale à zéro.

La fonction **Scan From String** est particulièrement utile si vous connaissez la longueur de l'en-tête. (`VOLTS DC` dans notre exemple), ou lorsque la chaîne de caractères ne contient que des caractères numériques valides.

2. Fermez le VI sans l'enregistrer, en choisissant **File»Close**.

E/S sur fichier

Les fonctions d'entrées/sorties (E/S) sur fichier de LabVIEW (**Functions»File I/O**) constituent un ensemble d'outils à la fois puissants et simples d'utilisation pour travailler avec des fichiers de données. En plus d'assurer la lecture et l'écriture des données, ces fonctions permettent également de déplacer et de renommer des fichiers et des répertoires, de créer des fichiers au format tableur avec du texte lisible au format ASCII, et d'écrire des données en format binaire pour gagner de l'espace et de la vitesse.

Vous pouvez stocker et extraire des données des fichiers selon trois formats différents.

- Format ASCII. Vous devez stocker des données dans le format ASCII lorsque vous prévoyez d'y accéder à partir d'un autre logiciel tel qu'un traitement de texte ou un tableur. Pour ce faire, vous devez convertir toutes les données en chaînes de caractères ASCII.
- Format Datalog. Ces fichiers au format binaire ne sont accessibles que par LabVIEW uniquement. Les fichiers Datalog de LabVIEW sont similaires aux fichiers de bases de données en ce sens qu'ils offrent la possibilité de stocker plusieurs types de données différents dans un seul enregistrement (log) de fichier.
- Format binaire. Avec ce format de fichiers, vous disposez de la méthode la plus compacte et la plus rapide pour stocker les données. Vous devez convertir les données en chaînes binaires et savoir précisément à quels types de données vous avez affaire pour sauvegarder et extraire les données de ces fichiers.

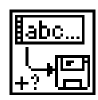
Cette section aborde en particulier les fichiers au format ASCII, qui reste le plus courant des formats. Veuillez vous reporter à la section intitulée *Quelques informations supplémentaires*, à la fin de ce chapitre, pour en savoir davantage sur les autres formats existants.

Pour consulter des exemples d'E/S sur fichier, reportez-vous au répertoire `examples\file`.

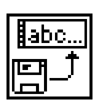
Les fonctions E/S sur fichier

La plupart des opérations d'E/S sur fichier supposent trois étapes de base, à savoir : l'ouverture d'un fichier existant ou la création d'un nouveau fichier, l'écriture ou la lecture du fichier, et enfin, la fermeture du fichier. A ce titre, LabVIEW contient de nombreux VIs utilitaires dans la palette **Functions»File I/O**. Dans cette section, nous allons aborder les neuf utilitaires de haut niveau. Ces fonctions s'articulent autour de VIs de niveau intermédiaire qui prévoient la gestion d'erreurs parmi les fonctions d'E/S sur fichier.

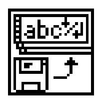
Vous pouvez également définir un séparateur ou une chaîne de séparateurs, comme les marques de tabulations, les virgules, etc. dans votre tableur. Cette opération vous évitera d'avoir à analyser le tableur si vous avez utilisé un séparateur différent du séparateur défini par défaut, en l'occurrence la marque de tabulation, lorsque vous avez configuré le tableur.



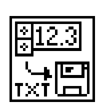
Le VI **Write Characters To File** écrit une chaîne de caractères dans un nouveau fichier ou l'ajoute dans un fichier existant. Ce VI ouvre ou crée le fichier, écrit les données puis ferme le fichier.



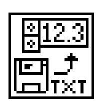
Le VI **Read Characters From File** lit un nombre précis de caractères dans un fichier à partir d'une position spécifiée. Ce VI ouvre au préalable le fichier puis le ferme.



Le VI **Read Lines From File** lit un nombre de lignes précis dans un fichier à partir d'une position spécifiée. Ce VI ouvre au préalable le fichier puis le ferme.



Le VI **Write To Spreadsheet File** convertit un tableau 1D ou 2D de nombres en simple précision en une chaîne de texte, puis écrit cette chaîne dans un nouveau fichier ou l'ajoute dans un fichier existant. Il est également possible de transposer les données. Ce VI ouvre ou crée au préalable le fichier puis le ferme. Vous pouvez utiliser ce VI pour créer des fichiers texte lisibles par la plupart des tableurs.



Le VI **Read From Spreadsheet File** lit un nombre précis de lignes ou de colonnes dans un fichier alphanumérique à partir d'une position spécifiée. Il convertit ensuite les données en un tableau 2D de nombres en simple précision. Vous pouvez également transposer le tableau. Ce VI ouvre au préalable le fichier puis le ferme. Vous pouvez utiliser ce VI pour lire des fichiers au format tableur sauvegardés au format texte.

Pour consulter d'autres fonctions d'E/S sur fichier, choisissez **Function»File I/O»Binary File VIs** ou **Function»File I/O»Advanced File Functions.**

Ecriture de données dans un fichier tableur

Une façon très courante de sauvegarder des données dans un fichier consiste à formater le fichier texte de façon à pouvoir l'ouvrir avec un tableur. Dans la plupart des tableurs, des tabulations séparent les colonnes et des caractères de fin de ligne (EOL) séparent les lignes, comme dans la figure suivante :

0.00➧0.4258¶
1.00➧0.3073¶
2.00➧0.9453¶
3.00➧0.9640¶
4.00➧0.9517¶

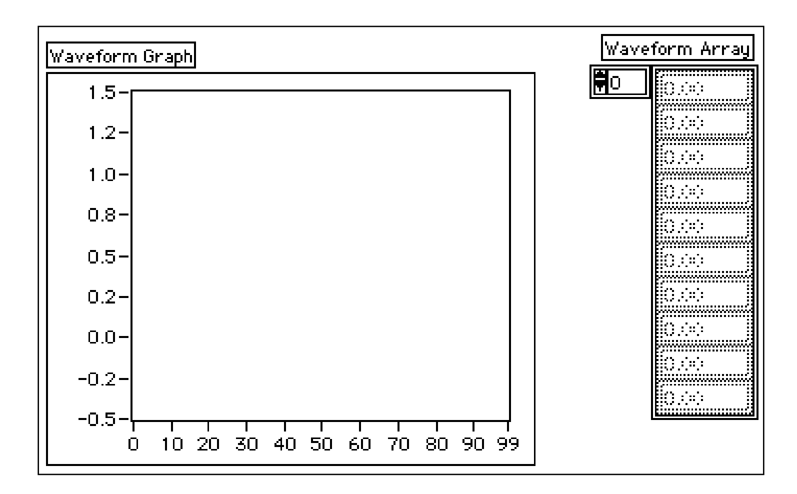
➧ = Tabulation
¶ = Séparateur de lignes

Lorsque vous ouvrez un fichier en utilisant un tableur, vous obtenez le tableau suivant :

	A	B	C
1	0	0.4258	
2	1	0.3073	
3	2	0.9453	
4	3	0.964	
5	4	0.9517	
6			

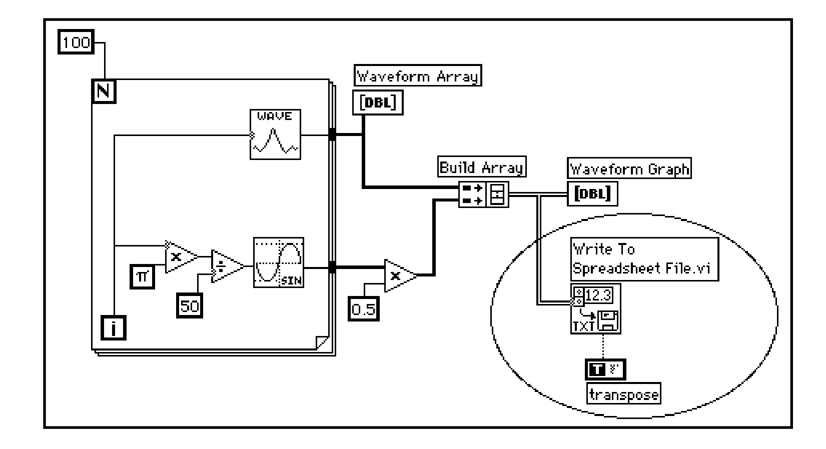
OBJECTIF Modifier un VI existant pour y intégrer une fonction d'E/S sur fichier, de façon à enregistrer les données dans un nouveau fichier au format ASCII. Vous pourrez alors accéder à ce fichier à partir d'un tableur.

La face-avant

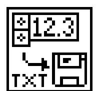


1. Ouvrez `My Graph Waveform Arrays.vi` que vous avez construit au chapitre 4 de ce tutorial. Ce VI génère deux tableaux de données qu'il trace sur un même graphe. Vous allez modifier ce VI pour écrire les deux tableaux dans un fichier où chaque colonne contient un tableau de données.

Le diagramme



2. Ouvrez le diagramme de `My Graph Waveform Arrays` puis modifiez le VI en y ajoutant les éléments présents dans l'ovale, comme indiqué ci-dessus.



Le VI **Write To Spreadsheet File** (**Functions»File I/O**) convertit le tableau à deux dimensions en chaîne de caractères au format tableur et l'écrit dans un fichier. Si vous n'avez pas indiqué le nom du chemin, une boîte de dialogue apparaît qui vous invite à entrer un nom de fichier. **Write To Spreadsheet File** écrit un tableau à une ou deux dimension(s) dans un fichier. Comme dans notre exemple, le tableau final étant déjà un tableau de données 2D, vous n'avez pas à le câbler à l'entrée 1D. Avec ce VI, vous pouvez utiliser un séparateur ou une chaîne de séparateurs de tableur, comme les marques de tabulation ou les virgules dans vos données.



Boolean Constant (**Functions»Boolean**). Cette constante permet de savoir si LabVIEW peut transposer ou non le tableau 2D avant de l'écrire dans le fichier. Pour que l'état de la valeur soit dans la situation TRUE, cliquez sur la constante à l'aide de l'outil Doigt. Dans ce cas, vous voulez que les données soient transposées parce que les tableaux de données sont relatifs aux lignes (chaque ligne du tableau à deux dimensions est un tableau de données). Comme chaque colonne du fichier tableur doit contenir un tableau de données, le tableau 2D doit être transposé en premier.

3. Retournez dans la face-avant et lancez le VI. Une fois les tableaux de données générés, une boîte de dialogue de fichier vous invite à donner le nom du nouveau fichier que vous êtes en train de créer. Entrez un nom de fichier puis cliquez sur **OK**.



Attention : ***n'essayez pas d'écrire des données dans des bibliothèques de VIs telles que*** `mywork.llb`***. Vous risqueriez d'écrire par dessus votre bibliothèque et donc de perdre tout votre travail.***

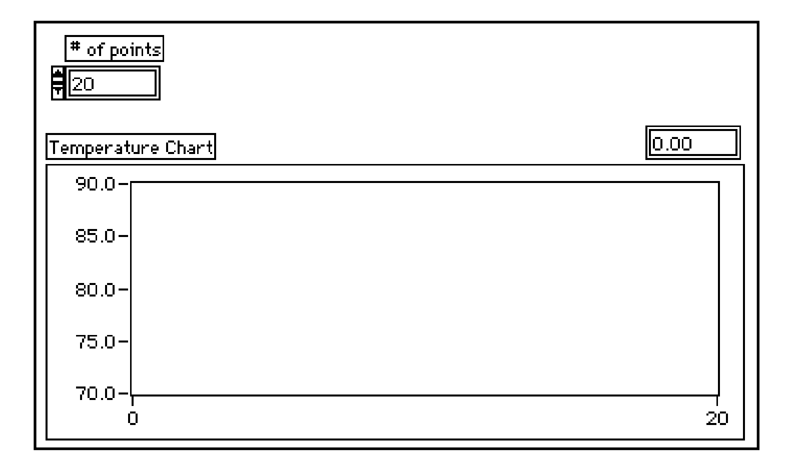
4. Enregistrez le VI sous le nom `My Waveform Arrays to File.vi`, puis fermez-le.
5. Vous pouvez dès à présent utiliser un tableur ou un éditeur de texte pour ouvrir et visualiser le fichier que vous venez de créer. Vous devriez donc voir apparaître deux colonnes de 100 éléments.

Dans cet exemple, les données ne sont pas converties ou écrites dans le fichier avant de disposer de l'intégralité des tableaux de données. Si vous êtes en train d'acquérir de gros paquets de données ou si vous souhaitez écrire les valeurs des données sur le disque au fur et à mesure de l'acquisition, alors il vous faudra utiliser un autre VI d'E/S sur fichier.

Ajout de données à un fichier

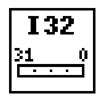
OBJECTIF Créer un VI pour ajouter des données de température à un fichier ASCII. Ce VI utilise une boucle *For* pour générer les valeurs de température et les stocker dans un fichier. Au cours de chaque itération, vous allez convertir les données en chaîne de caractères, ajouter une virgule comme séparateur, puis ajouter la chaîne de caractères dans un fichier.

La face-avant



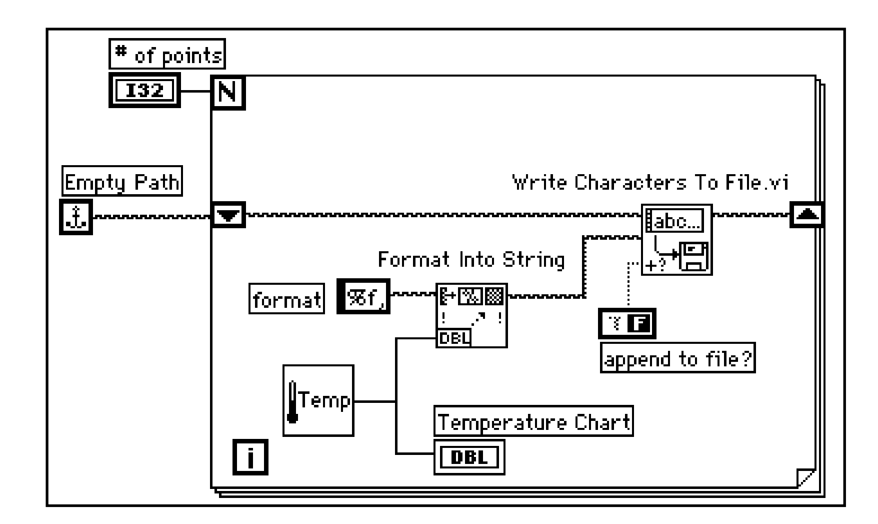
1. Ouvrez une nouvelle face-avant et placez-y les objets comme indiqué ci-dessus.

Cette face-avant contient une commande numérique et un graphe déroulant. Choisissez l'option **Show»Digital Display**. La commande `# of points` indique le nombre de valeurs de température à acquérir et à écrire dans le fichier. Le graphe déroulant restitue la courbe de température. Modifiez l'échelle de l'axe des Y comme suit, de 70.0 à 90.0, et celle de l'axe des X comme suit, de 0 à 20.



2. Ouvrez un menu local sur la commande numérique `# of points` et choisissez **Representation»Long**.

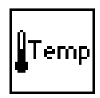
Le diagramme



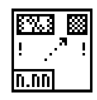
1. Ouvrez le diagramme.
2. Ajoutez la boucle *For* et agrandissez-la. Ce VI génère le nombre de valeurs de température spécifié par la commande # `of Points`.
3. Ajoutez un registre à décalage sur la boucle en ouvrant un menu local sur sa bordure. Le registre à décalage contient le nom du chemin d'accès au fichier.
4. Finissez de câbler les objets.



Constante **Empty Path** (**Functions»File I\O»File Constants**). Cette fonction permet d'initialiser le registre à décalage de sorte qu'à la première tentative d'écriture dans le fichier, le chemin d'accès soit vide. Une boîte de dialogue de fichier vous invite alors à entrer le nom de fichier.



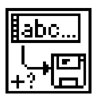
Le VI **My Thermometer** que vous avez construit au chapitre 2 (**Functions»Select a VI...)** ou le VI **Digital Thermometer** (**Functions»Tutorial**) simule un relevé de température réalisé par un capteur de température.



La fonction **Format Into String** (**Functions»String**) convertit la mesure de température (un nombre) en chaîne de caractères et effectue la concaténation de la virgule qui suit.



Constante **String Constant** (**Functions»String**). Ce format de chaîne indique que vous souhaitez convertir un nombre en une chaîne au format fractionnaire, suivie d'une virgule.



Le VI **Write Characters To File** (**Functions»File I/O)** écrit une chaîne de caractères dans un fichier.



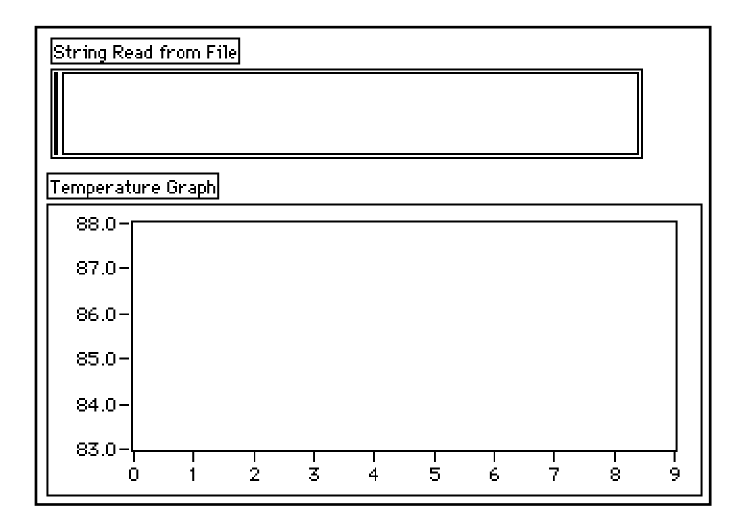
Boolean Constant (**Functions**»**Boolean**) règle l'entrée `append to file`? du VI **Write Characters To File VI** sur la valeur TRUE de sorte que les nouvelles valeurs de température acquises lors de l'itération de la boucle soient ajoutées au fichier sélectionné. Avec l'outil Doigt, cliquez sur la constante pour régler sa valeur sur TRUE.

5. Retournez dans la face-avant et exécutez le VI en réglant le paramètre `# of points` sur 20. Une boîte de dialogue de fichier vous invite à entrer un nom de fichier. Lorsque c'est chose faite de fichier, le VI commence à écrire les valeurs de température dans celui-ci au fur et à mesure qu'il génère les points.
6. Enregistrez le VI sous le nom `My Write Temperature to File.vi`, puis fermez-le.
7. Utilisez n'importe quel logiciel de traitement de texte comme *Write* pour Windows, *Teach Text* pour Macintosh, et *Text Editor in Open Windows* pour UNIX pour ouvrir ce fichier de données et étudier son contenu. Vous devriez obtenir un fichier de vingt valeurs (avec trois chiffres après la virgule) séparées par des virgules.

Lecture de données en provenance d'un fichier

OBJECTIF Créer un VI capable de lire les données que vous venez d'écrire dans l'exemple précédent et d'afficher les données sous la forme d'un graphe oscilloscopique. Attention : vous devez lire les données dans le même format que celui dans lequel vous les avez enregistrées. Par conséquent, puisque vous les avez sauvegardées au départ dans un format ASCII en utilisant des données du type chaînes de caractères, vous devez les retrouver en chaîne lors de la lecture avec un des VIs d'E/S sur fichier.

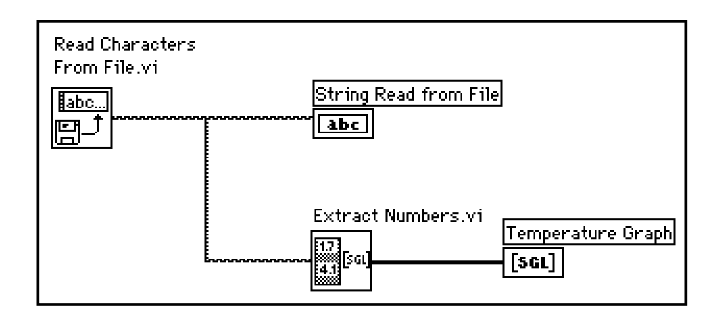
La face-avant



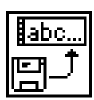
1. Ouvrez une nouvelle fenêtre de face-avant et construisez la face-avant reproduite ci-dessus.

La face-avant contient un indicateur de chaîne de caractères et un graphe oscilloscopique. L'indicateur **String Read from File** affiche les valeurs de températures séparées par des virgules, obtenues dans l'exemple précédent. Le graphe oscilloscopique restitue la courbe de température.

Le diagramme

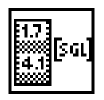


1. Construisez le diagramme reproduit ci-dessus.



Le VI **Read Characters From File** (**Functions»File I/O**) lit les données du fichier et restitue les informations obtenues sous forme de chaîne de caractères. Si vous n'avez pas spécifié de nom de chemin, alors une boîte de dialogue vous invite à entrer un nom de fichier. Dans cet exemple, vous n'avez pas besoin d'indiquer le nombre de caractères à lire puisque celui qui figure dans le fichier est inférieur à la quantité prévue par défaut, soit 512.

Vous devez savoir de quelle manière les données ont été stockées dans un fichier pour pouvoir les relire. Si vous connaissez la longueur d'un fichier, vous pouvez utiliser le VI **Read Characters From File** pour spécifier le nombre de caractères à lire.



Le VI **Extract Numbers** (`examples\general\strings.llb`) prend une chaîne de caractères ASCII contenant des nombres séparés par des virgules, des retours à la ligne ou tout autre caractère non numérique, puis la convertit en un tableau de valeurs numériques.

2. Retournez dans la face-avant et lancez le VI. Choisissez le fichier de données que vous venez d'écrire sur le disque lorsque la boîte de dialogue vous le demande. Vous devriez obtenir les mêmes valeurs affichées sur le graphe que celles de l'exemple du VI **My Write Temperature to File**.
3. Enregistrez le VI sous le nom `My Temperature from File.vi`, puis fermez-le.

Mise en œuvre des fonctions d'E/S sur fichier

Il arrive parfois que les fonctions d'E/S sur fichier ne fournissent pas les fonctionnalités attendues pour enregistrer les données sur disque. Le cas échéant, vous devez utiliser les fonctions du menu **Functions»File I/O»Advanced**.

Spécification d'un fichier

Il existe deux manières de spécifier un fichier, soit par programmation, soit par l'intermédiaire d'une boîte de dialogue. Avec la méthode de programmation, vous indiquez le nom du fichier et le nom du chemin d'accès.

(Windows) Un nom de chemin se compose du nom du lecteur, par exemple C, suivi du signe de ponctuation deux points (:), suivis des noms de répertoires séparés par une barre oblique inverse, suivis du nom de fichier. Citons à titre d'exemple, `C:\DATADIR\TEST1` où

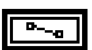
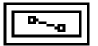
`TEST1` représente le nom de fichier qui se trouve dans le répertoire `DATADIR` sur le lecteur C.

(Macintosh) Un nom de chemin se compose du nom du lecteur, suivi du signe de ponctuation deux points (:), suivis par des noms de dossiers séparés les uns des autres par deux points, suivis du nom du fichier. Citons à titre d'exemple, `HardDrive:DataFolder:Test1` où `Test1` représente le nom du fichier qui se trouve dans le dossier `DataFolder` du disque dur (*hard drive*).

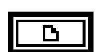
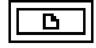
(UNIX) Un nom de chemin se compose des noms de répertoire séparés les uns des autres par une barre oblique, suivis d'un nom de fichier. Citons à titre d'exemple, `/usr/datadirectory/test1` où `test1` représente le nom de fichier dans le répertoire `/usr/datadirectory`.

(Toutes les plates-formes) En utilisant la méthode de la boîte de dialogue, la fonction File Dialog affiche une boîte de dialogue que vous pouvez utiliser pour rechercher de manière interactive un répertoire et y entrer le nom du fichier.

Chemins d'accès et numéros de référence

Un chemin est un type de données de LabVIEW qui sert à identifier des fichiers. Vous pouvez entrer ou afficher un chemin de fichier en utilisant une syntaxe standard, laquelle dépend de la plate-forme utilisée, avec la commande du chemin et son indicateur. Dans la plupart des cas, la commande et l'indicateur du chemin s'utilisent comme une commande ou un indicateur de chaîne de caractères, à cette différence près que LabVIEW formate le chemin de façon à ce qu'il soit reconnu par la machine sur laquelle vous travaillez.

Un numéro de référence est un type de données utilisé par LabVIEW pour identifier les fichiers ouverts. Lorsque vous ouvrez un fichier, LabVIEW renvoie un numéro de référence associé au fichier ouvert. Toutes les opérations effectuées sur des fichiers ouverts utilisent les numéros de référence afin d'identifier chacun de ces fichiers. Le numéro de référence reste valide tant que le fichier reste ouvert. Si vous le fermez, LabVIEW élimine le numéro de référence associé au fichier fermé. Si vous ouvrez à nouveau le fichier, un nouveau numéro de référence lui sera attribué, différent du premier.

LabVIEW ne se limite pas à associer un numéro de référence à un fichier, il mémorise également les informations relatives à chaque

numéro de référence utilisé, comme l'emplacement du fichier à lire et son niveau d'accessibilité pour les autres utilisateurs. Il est ainsi possible d'effectuer plusieurs opérations en parallèle, indépendantes les unes des autres, sur un même fichier. Si vous ouvrez un fichier à plusieurs reprises, il se verra à chaque fois associé un numéro de référence différent.

Les fonctions d'E/S sur fichier ne contiennent pas de gestion d'erreur et ne renvoient donc pas de messages d'erreur. Par conséquent, si vous construisez des applications qui utilisent des fonctions de bas niveau, vous devez gérer vous-même la gestion d'erreur et la manipulation des messages associés afin d'éviter tout problème.

Exemples d'E/S sur fichier

Vous pouvez vous reporter aux exemples suivants pour voir comment utiliser les fonctions E/S sur fichier avec leurs différentes techniques de gestion d'erreurs :

Le VI **Write to Text File** (dans `examples\file\smplfile.llb`) écrit un fichier texte ASCII qui contient des valeurs de données horodatées.

Le VI **Read from Text File** (dans `examples\file\smplfile.llb`) lit un fichier texte ASCII qui contient des valeurs de données horodatées.

Résumé

Une chaîne de cractères est un ensemble de caractères ASCII. Les commandes et les indicateurs de chaînes de caractères se trouvent dans la palette **Controls»String & Table**.

LabVIEW offre de nombreuses fonctions permettant de manipuler les chaînes de caractères. Toutes ces fonctions se trouvent dans la palette **Functions»String**.

LabVIEW peut effectuer des opérations sur les fichiers. Pour écrire dans un fichier, vous créez un nouveau fichier (ou bien ouvrez un fichier existant), écrivez les données puis fermez le fichier. De même, pour lire un fichier, vous ouvrez un fichier existant, lisez les données, puis fermez le fichier. Quant aux opérations courantes d'E/S sur fichier, les VIs réunissent toutes ces opérations en un seul sous-VI.

Pour plus de souplesse, vous pouvez utiliser les VIs utilitaires de niveau intermédiaire ou les fonctions d'E/S sur fichier.

Retenez que vous pouvez utiliser des chaînes de caractères ou d'autres types de données pour vos opérations d'entrée et de sortie sur fichier. Si le VI écrit des chaînes de caractères, il forme un fichier ASCII, alors qu'avec d'autres formats de données, il génère un fichier au format binaire. Les fichiers binaires sont bien plus rapides et plus compacts. En revanche, les fichiers ASCII sont lisibles par de nombreux logiciels. Nous vous renvoyons aux informations fournies ultérieurement dans ce chapitre pour savoir comment créer et lire des fichiers de données binaires.

N'oubliez pas d'utiliser la gestion d'erreurs lorsque vous écrivez ou lisez des données sur des fichiers. Vous gagnerez ainsi beaucoup de temps en détectant les valeurs de sortie erronées survenant lors de l'utilisation des fonctions d'E/S sur fichier.

Quelques informations supplémentaires

Les fichiers Datalog

Les exemples présentés dans ce chapitre reproduisent des méthodes simples pour travailler avec des fichiers contenant des données stockées sous la forme d'une séquence de caractères ASCII. Cette approche est identique lorsque vous créez des fichiers qui seront lus par d'autres logiciels tels que des tableurs. LabVIEW offre également un autre format de fichier appelé *datalog file*. Ce format de fichier permet de stocker des données sous la forme d'une séquence d'enregistre-ments d'un seul type de données arbitraires que vous déterminez à la création du fichier. LabVIEW indexe les données de ces enregistrements dans un fichier Datalog. Si, dans un fichier Datalog, les enregistrements doivent tous être du même type, ce type peut être très complexe. Par exemple, vous pouvez spécifier que chaque enregistrement est un *cluster* composé d'une chaîne de caractères, d'un nombre et d'un tableau.

Si vous récupérez les données avec un VI de LabVIEW, vous pouvez éviter d'écrire ces données dans des fichiers au format ASCII, dans la mesure où la conversion de données en chaînes de caractères risque de prendre du temps. Par exemple, la conversion d'un tableau à deux dimensions en une chaîne de caractères dans un format tableur avec des

en-têtes et des repères de temps, est une opération plutôt complexe. Si vous pouvez éviter de stocker les données dans un format lisible par d'autres logiciels, vous avez tout intérêt à écrire vos données dans un format type Datalog. Ainsi, l'écriture des données dans un fichier ne nécessite que peu de manipulation, ce qui accélère considérablement les opérations d'écriture et de lecture. La récupération de données s'en trouve aussi simplifiée, puisque vous pouvez lire les blocs de données originaux comme une simple succession d'enregistrements, sans avoir à connaître le nombre d'octets présents dans chaque enregistrement. LabVIEW enregistre la totalité des données de chaque enregistrement d'un fichier Datalog.

Le VI **Write to Datalog File** (dans `examples\file\datalog.llb`) crée un nouveau fichier Datalog et écrit le nombre d'enregistrements spécifié dans le fichier. Chaque enregistrement est un *cluster* avec une chaîne de caractères et un tableau de nombres en simple précision.

Pour pouvoir lire un fichier Datalog, vous devez avoir le même type de données que celui que vous avez utilisé lors de l'écriture. Le VI **Read from Datalog File** (dans `examples\file\datalog.llb`) lit un fichier Datalog créé par le VI **Write to Datalog File**. La lecture de l'enregistrement est un cluster contenant une chaîne de caractères et un tableau de nombres en simple précision.

Les fichiers de communication de données binaires

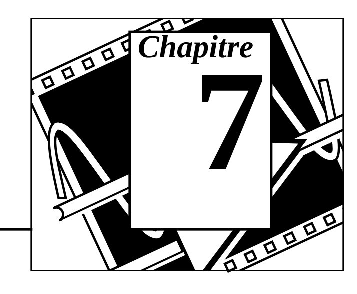
L'écriture des données dans des fichiers binaires peut être plus rapide et prendre moins d'espace mémoire que les fichiers de communication ASCII. Cela étant, cette approche est plus complexe car vous devez porter une attention toute particulière au format du fichier et reconstruire les données d'origine. Si le fichier contient différents éléments constitués de données écrites dans un format binaire, et que vous voulez récupérer le format d'origine, vous devez prévoir dans le fichier un en-tête décrivant la structure des données. Par exemple, avant d'écrire un tableau de données numériques dans un fichier, vous connaîtrez grâce à cet en-tête la quantité de données nécessaires à la reconstruction des éléments d'origine.

L'exemple `Binary vs ASCII` dans `examples\general\strings.llb` illustre ce qui différencie une chaîne binaire d'une chaîne ASCII.

Veuillez vous reporter aux VIs **Write To I16 File**, **Read From I16 File**, **Write To SGL File**, et **Read From SGL File** (**Functions»File I/O» Binary**) pour consulter d'autres exemples d'écriture et de lecture de fichiers binaires.

E/S d'erreurs des fonctions d'E/S sur fichier

Les fonctions d'E/S sur fichier contiennent également des *clusters* de gestion d'erreurs d'E/S, dont le rôle consiste à détecter les erreurs qui surviennent en entrée et en sortie. Avec de tels *clusters*, vous pouvez chaîner plusieurs fonctions les unes aux autres. Lorsqu'une erreur survient dans une fonction, cette fonction ne s'exécute pas et transmet l'erreur à la fonction suivante. Pour plus d'informations sur la détection des erreurs d'E/S, veuillez consulter **Online Reference»Function and VI Reference»Error I/O in File I/O Functions**.





La personnalisation des VIs

Vous allez apprendre :

- A utiliser l'option **VI Setup...**.
- A utiliser l'option **SubVI Node Setup...**.
- A personnaliser les commandes et les indicateurs.

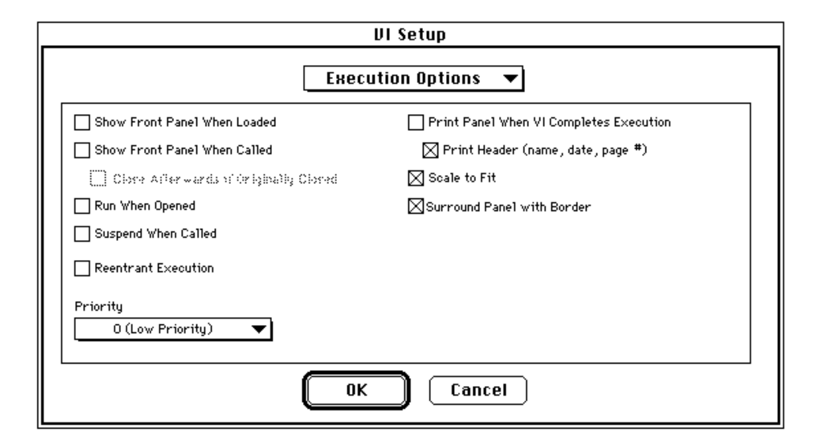
Pour consulter des exemples de VIs personnalisés, veuillez vous reporter à `examples\general\viopts.llb.`

La configuration d'un VI

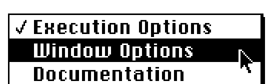
Il existe différentes manières de configurer l'exécution de vos VIs. Pour accéder à ces différentes configurations, ouvrez un menu local sur le cadre icône situé en haut à droite de la face-avant, puis choisissez l'option **VI Setup...**.



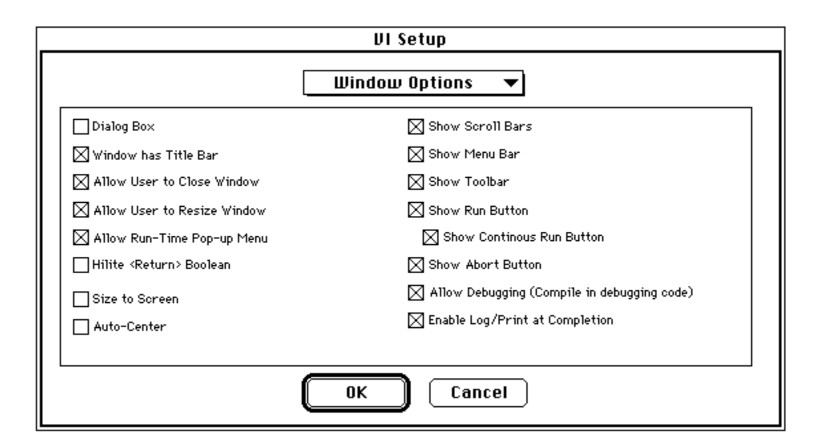
Comme le montre l'illustration suivante, une boîte de dialogue intitulée **VI Setup** apparaît, dans laquelle figurent toutes les options de configuration possibles. Toutes ces options sont reprises en détail au chapitre 6, *La création de faces-avant locales et la définition du fenêtrage*, du *Manuel de l'utilisateur LabVIEW*.



Les options de fenêtrage

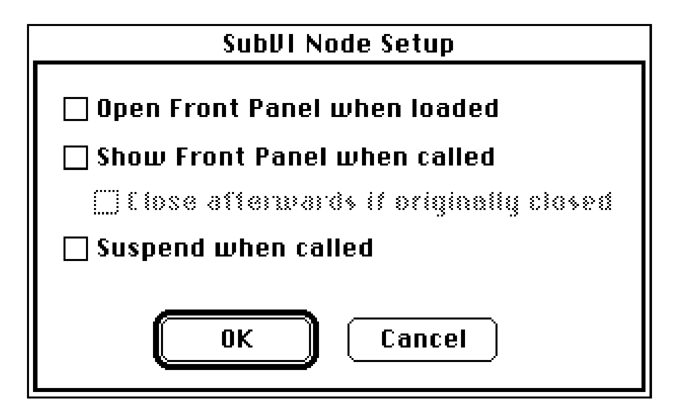


Les options **Window Options** contrôlent la présentation des VIs en cours d'exécution. Pour passer d'**Execution Options** à **Window Options**, cliquez sur la flèche orientée vers le bas qui se trouve dans la barre de menus.



La configuration de nœud de sous-VI

Il existe plusieurs options de configuration des sous-VIs que vous pouvez modifier. Toutes ces options sont accessibles en cliquant sur l'icône du sous-VI (dans le diagramme du VI appelant), et en choisissant l'option **SubVI Node Setup...**. L'illustration ci-après représente la boîte de dialogue **SubVI Node Setup**.



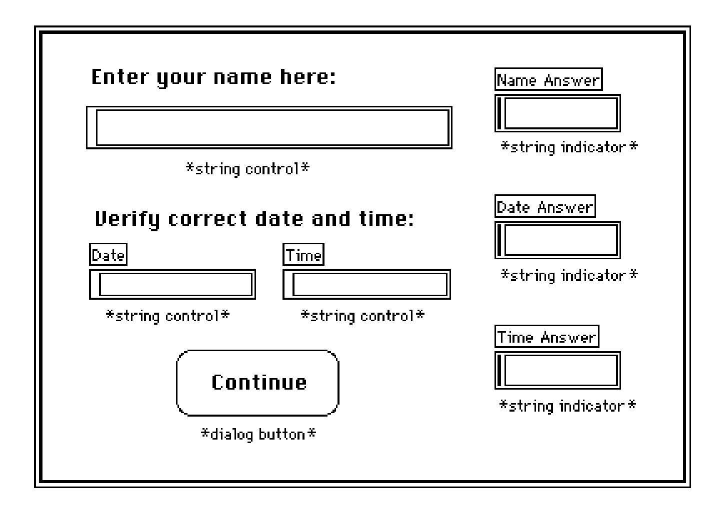
☞ **Remarque :** ***si vous choisissez une option dans la boîte de dialogue VI Setup... d'un VI, l'option choisie s'applique alors à toutes les instances du VI. En revanche, si vous choisissez une option dans la boîte de dialogue*** **SubVI Node Setup*****, l'option choisie ne s'appliquera qu'au nœud en question.***

La mise en œuvre des options de configuration d'un sous-VI

OBJECTIF Construire un VI qui simule une application de test. Au démarrage du VI, un autre sous-VI ouvre une nouvelle face-avant qui invite l'utilisateur à entrer son nom et à vérifier la date et l'heure. La face-avant reste ouverte jusqu'à ce que l'utilisateur clique sur le bouton **Continue**.

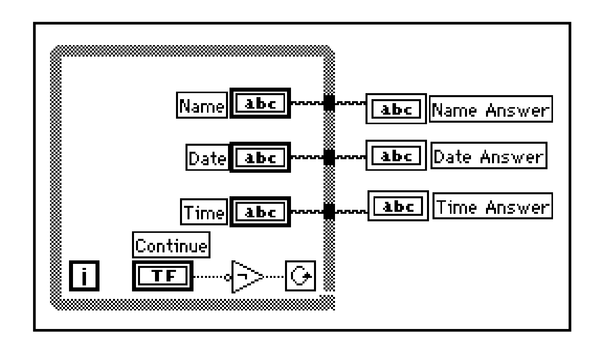
Vous devez commencer par construire le VI qui ouvrira sa propre face-avant, vous demandera les informations, et attendra que vous cliquiez sur le bouton booléen. Vous pourrez ensuite utiliser ce VI comme sous-VI dans le diagramme du VI principal.

La face-avant

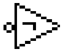


1. Ouvrez une nouvelle fenêtre et construisez la face-avant reproduite ci-dessus.

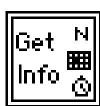
Le diagramme



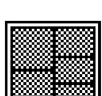
1. Construisez le diagramme de l'illustration précédente.



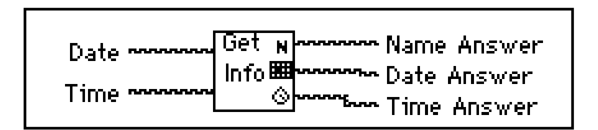
Fonction **Not** (**Functions»Boolean**). Dans cet exercice, la fonction **Not** inverse la valeur du bouton **Continue** de manière à ce que la boucle *While* s'exécute indéfiniment jusqu'à ce que vous cliquiez sur le bouton **Continue**. (La valeur par défaut du bouton est FALSE.)



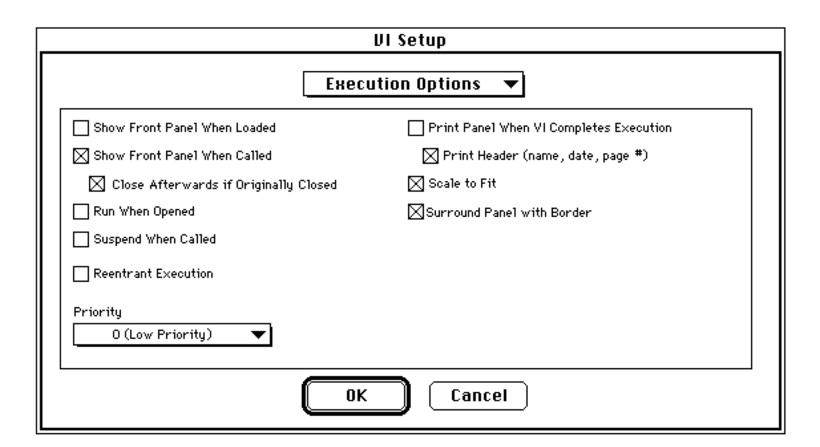
2. Créez l'icône du VI présenté à gauche. Pour accéder à l'Editeur d'icônes, ouvrez un menu local sur le cadre icône de la face-avant puis choisissez **Edit Icon**.
3. Pour passer dans le cadre connecteur, ouvrez un menu local sur le cadre icône puis choisissez l'option **Show Connector**.



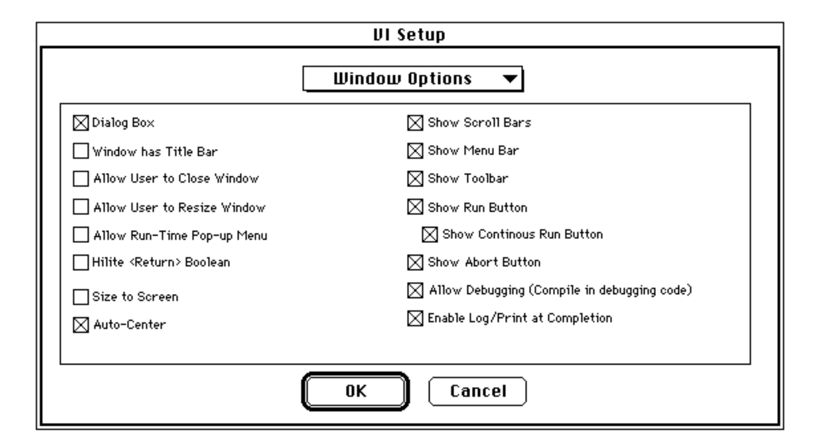
4. Construisez le connecteur. Pendant sa construction, vous remarquerez que le cadre connecteur par défaut n'est pas celui reproduit dans la partie gauche de l'illustration. Pour obtenir le cadre connecteur qui convient, choisissez **Patterns** dans le menu local du connecteur. Choisissez le modèle à deux entrées et à trois sorties. Enfin, choisissez **Flip Horizontal**. Maintenant vous pouvez connecter les commandes **Date** et **Time** aux deux connecteurs situés à gauche de l'icône, et les indicateurs restituant **Name Answer**, **Date Answer** et **Time Answer** aux trois connecteurs situés à droite de l'icône conformément à la fenêtre d'aide suivante. Une fois le connecteur créé, retournez à l'afficheur de l'icône.



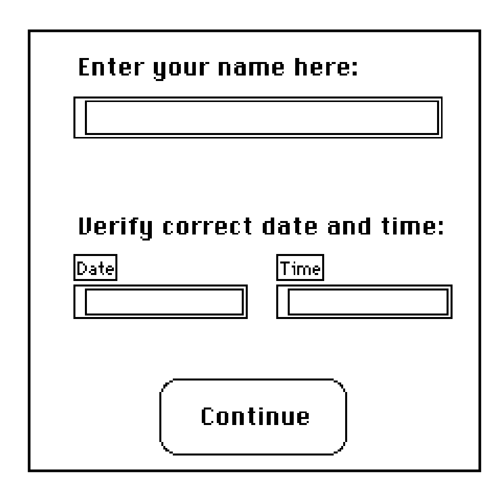
5. Enregistrez le VI sous le nom `My Get Operator Info.vi`. Vous pouvez maintenant personnaliser le VI en utilisant les options de configuration pour qu'il ressemble à une boîte de dialogue.



Configurez les options **Execution Options** en retenant tous les paramètres sélectionnés dans l'illustration précédente. Puis, modifiez les options **Window Options** pour obtenir une configuration semblable à celle de l'illustration suivante.

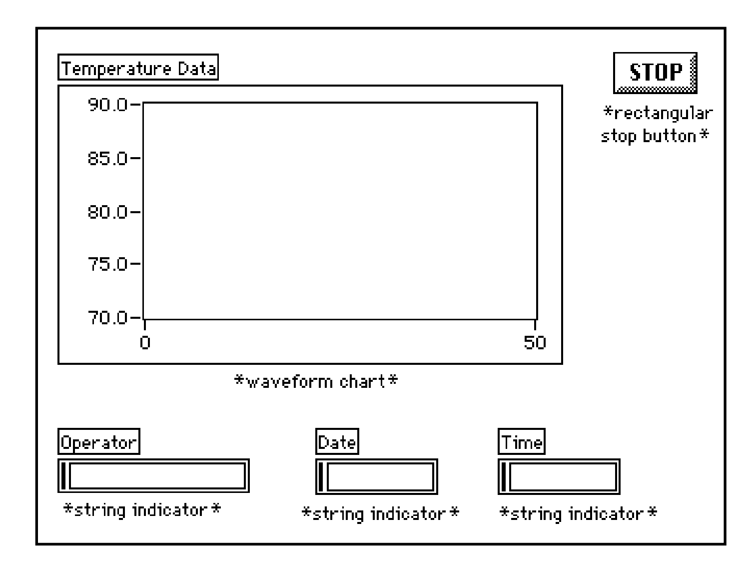


6. Lorsque vous en aurez terminé avec la configuration du VI, modifiez la taille de la face-avant comme dans l'illustration suivante de manière à masquer les trois indicateurs de chaînes de caractères.

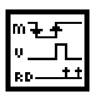


7. Enregistrez le VI et fermez-le. Vous utiliserez bientôt ce VI comme sous-VI.

La face-avant



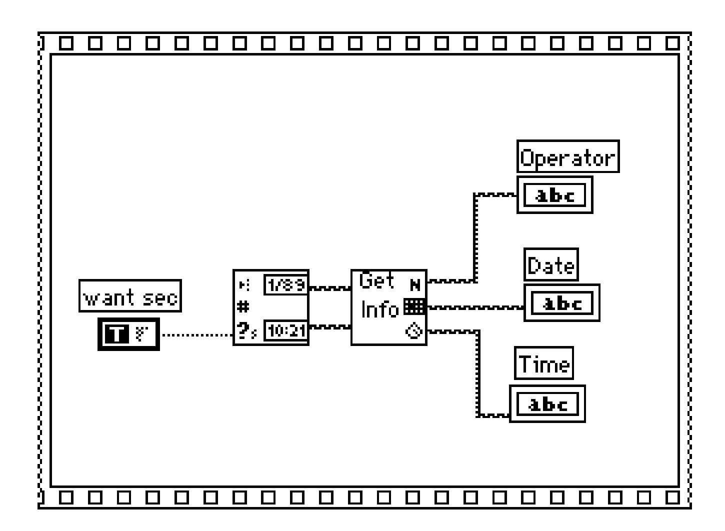
1. Ouvrez une nouvelle face-avant.
2. Placez un graphe déroulant (**Controls»Graph**) dans la face-avant et nommez-le `Temperature Data`.
3. Modifiez l'échelle du graphe de telle sorte que les seuils haut et bas soient respectivement réglés sur 90.0 et 70.0.
4. Terminez la construction de la face-avant conformément à l'illustration précédente.



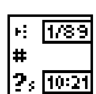
5. Ouvrez un menu local sur le bouton rectangulaire libellé STOP, puis choisissez **Mechanical Action»Latch When Released.**

Le graphe déroulant affiche la température au fur et à mesure de son acquisition.

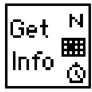
Le diagramme



6. Construisez le diagramme présenté ci-dessus.
7. Ajoutez une structure Séquence et les éléments suivants dans le cadre 0.



La fonction **Get Date/Time String** (**Functions»Time & Dialog**) donne la date et l'heure en cours.

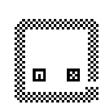


Le VI **Get Operator Info** (**Functions»Select a VI...**) ouvre une face-avant permettant à l'utilisateur de saisir son nom, la date et l'heure.



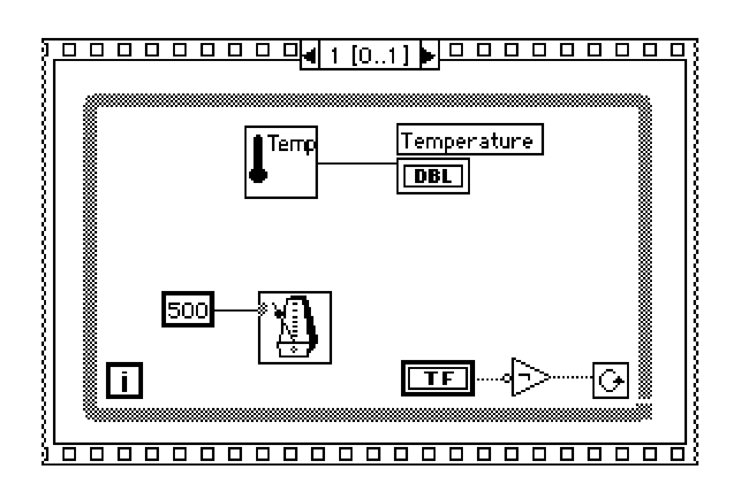
Boolean Constant (**Functions»Boolean**) vérifie si la date d'entrée et la chaîne heure sont vraies. Pour régler cette option sur TRUE, cliquez sur la constante avec l'outil Doigt.

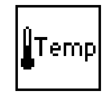
8. Ouvrez un menu local sur la structure Séquence puis choisissez l'option **Add Frame After** dans le menu local.



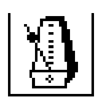
9. Placez une boucle *While* dans le cadre 1 de la structure Séquence.

10. Ajoutez tous les objets de l'illustration ci-dessous, dans le cadre 1.





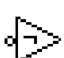
Le VI **Digital Thermometer** (**Functions»Tutorial**) fournit une mesure de température provenant d'un simulateur de capteur de température ou du VI **My Thermometer** (**Functions»Select a VI...**) que vous avez construit au chapitre 2.



La fonction **Wait Until Next ms Multiple** (**Functions»Time & Dialog**) déclenche l'exécution de la boucle *For* en ms.

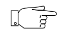


Numeric Constant (**Functions»Numeric**). Vous pouvez également ouvrir un menu local sur la fonction **Wait Until Next Tick Multiple** et choisir **Create Constant** pour créer et câbler automatiquement la constante numérique. La constante numérique prolonge le temps d'exécution de la boucle qui s'exécute toutes les 500 ms (0,5 secondes).



La fonction **Not** (**Functions»Boolean**) inverse la valeur du bouton STOP de telle sorte que la boucle *While* s'exécute indéfiniment jusqu'à ce que vous cliquiez sur le bouton STOP.

11. Enregistrez le VI sous le nom `My Pop-up Panel Demo.vi`.
12. Lancez le VI. La face-avant du VI **Get Operator Info** s'ouvre et vous invite à saisir votre nom, ainsi que la date et l'heure. Cliquez sur le bouton **Continue** pour revenir au VI appelant. Les données de température sont ensuite acquises jusqu'à ce que vous cliquiez sur le bouton STOP.



Remarque : ***la face-avant du* VI Get Operator Info *s'ouvre du fait des options choisies dans la boîte de dialogue* VI Setup.... *N'essayez pas d'ouvrir la face-avant du sous-VI à partir du diagramme du VI* My Pop-up Panel Demo.**

13. Fermez toutes les fenêtres.

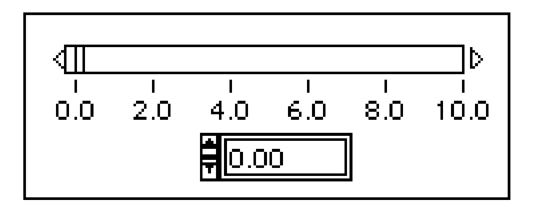
Les indicateurs et commandes personnalisés

LabVIEW dispose d'un éditeur de commandes qui vous permet de personnaliser les commandes de vos faces-avant. Il sert aussi à enregistrer une commande personnalisée pour que vous puissiez l'utiliser dans d'autres VIs.



Pour appeler cet éditeur de commandes, sélectionnez une commande à l'aide de l'outil Flèche, puis choisissez **Edit»Edit Control**.

Lorsque vous éditez une commande, une nouvelle fenêtre s'ouvre avec une copie de la commande en question.



Vous pouvez alors personnaliser la commande en la coloriant, en modifiant ses dimensions, en ajoutant de nouveaux éléments aux *clusters*, etc. Toutes ces modifications n'affectent en rien le VI original tant que vous n'avez pas sélectionné l'option **File»Apply Changes**, ou que vous n'avez pas fermé la fenêtre et répondu YES à la question sur le remplacement de la commande de départ.

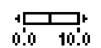
Si vous souhaitez utiliser la commande dans d'autres VIs, vous pouvez l'enregistrer en tant que *commande personnalisée* en choisissant **File»Save**. Après l'avoir enregistrée, vous pouvez la placer dans d'autres faces-avant en utilisant l'option **Controls»Select a Control...**.

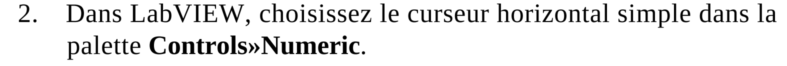
Vous avez également la possibilité d'importer des images à partir d'un programme de dessin ou de peinture dans une commande ou un indicateur. Imaginez, par exemple que vous ayez une commande curseur horizontale et que vous vouliez que ce curseur ressemble à une

boîte et la glissière à un tapis roulant. Vous pouvez charger un exemple de cette commande en ouvrant un menu local sur une face-avant, en choisissant l'option **Controls»Select a Control...** et en ouvrant `examples\general\controls\custom.llb\box on belt`.

1. ***(Windows et Macintosh)*** Utilisez un programme graphique pour dessiner une boîte puis importez cette image dans LabVIEW au moyen du presse-papiers.

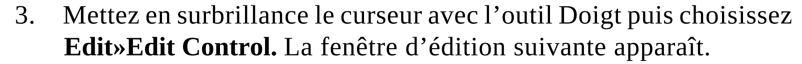
   (UNIX) Utilisez un programme graphique pour dessiner une boîte, puis enregistrez l'image sous un fichier *xwd (X Window Dump)*. Importez cette image dans le presse-papiers de LabVIEW en choisissant **File»Import Picture...**.



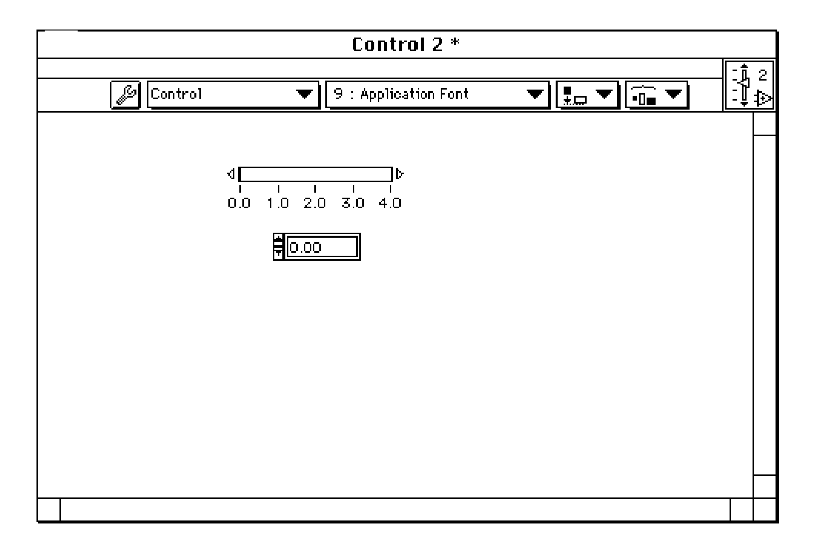


2. Dans LabVIEW, choisissez le curseur horizontal simple dans la palette **Controls»Numeric**.

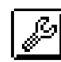




3. Mettez en surbrillance le curseur avec l'outil Doigt puis choisissez **Edit»Edit Control.** La fenêtre d'édition suivante apparaît.

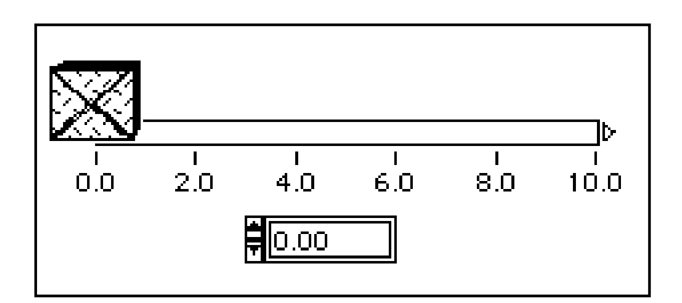


4. Servez-vous de l'outil Flèche pour choisir le curseur.

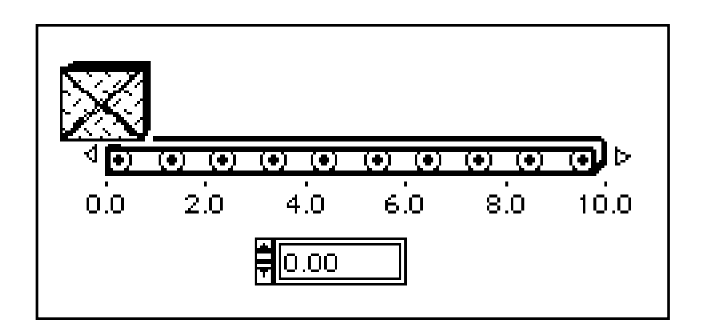


5. Cliquez sur la clé, représentée à gauche, dans la barre d'outils de la fenêtre d'édition.

6. Ouvrez un menu local sur le curseur horizontal puis choisissez **Import Picture** pour recopier l'image sur le curseur. Cette opération n'est possible que si l'image se trouve dans le presse-papiers de LabVIEW. Reportez-vous à l'étape 1 pour plus d'informations sur la façon d'importer une image dans le presse-papiers. La boîte devrait alors remplacer le curseur, comme dans l'illustration suivante.



7. Maintenant, dessinez le tapis roulant et importez-le dans LabVIEW comme dans l'étape 1. Revenez à la fenêtre d'édition et recommencez les étapes numérotées de trois à cinq. Le curseur horizontal devrait maintenant ressembler au curseur présenté dans l'illustration suivante.



Vous pouvez enregistrer ce curseur afin de pouvoir l'utiliser dans d'autres VIs. Pour en savoir plus sur l'éditeur de commandes, veuillez vous reporter au chapitre 23, *Les commandes personnalisées et les définitions de types,* du *Manuel de l'utilisateur LabVIEW.*

Pour consulter des exemples de commandes personnalisées, veuillez vous reporter au répertoire `examples\general\control`.

Résumé

En utilisant les options de configuration, vous pouvez modifier les caractéristiques d'exécution des VIs. Vous pouvez ainsi masquer les boutons de la barre d'outils, exécuter le VI dès qu'il est chargé, ouvrir des faces-avant quand elles sont appelées, etc. Pour ce faire, vous disposez de deux boîtes de dialogue différentes : la boîte de dialogue **VI Setup** et la boîte **SubVI Node Setup**.

Toutes les caractéristiques d'exécution d'un VI qui sont modifiées à partir de la boîte de dialogue **VI Setup** affectent *toutes* les exécutions de ce VI en tant que VI principal ou en tant que sous-VI. En revanche, les modifications effectuées à partir de la boîte de dialogue **SubVI Node Setup** n'affectent que la copie du VI utilisée comme sous-VI dans le nœud concerné. Les autres nœuds du VI ne s'en trouvent pas affectés.

La boîte de dialogue **VI Setup** offre également des options permettant de masquer les boutons de la barre d'outils. Ces options sont particulièrement utiles lors de la construction de VIs mis en œuvre par d'autres opérateurs, ou pour construire des VIs pour des systèmes de test plus complexes.

Vous pouvez concevoir vos propres commandes ou indicateurs en utilisant l'éditeur de commandes. Vous pouvez même importer vos images personnalisées dans les commandes en utilisant l'éditeur de commandes.

Quelques informations supplémentaires

La simulation d'une commande ou d'un indicateur

Dans LabVIEW, un objet de face-avant peut être une commande ou un indicateur. Dans certains cas, vous pouvez avoir besoin d'une commande qui se comporte à la fois comme une commande et comme un indicateur. C'est ce que tentait d'illustrer l'exemple précédent avec la date et l'heure. Vous voulez afficher la date et l'heure en cours mais il arrive parfois que l'horloge interne de l'ordinateur ne soit pas à l'heure. Le cas échéant, on a besoin d'une commande et d'un indicateur : l'indicateur pour corriger la date et l'heure et la commande pour pouvoir modifier les valeurs affichées en cas d'erreur.

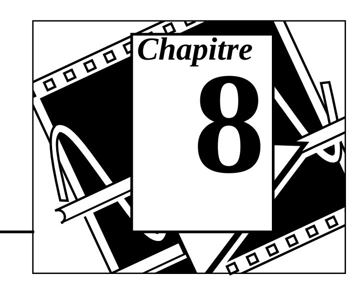
C’est l’objet du VI **My Pop-Up Panel**. L’indicateur du programme principal devient la commande de la face-avant. Vous pouvez ainsi corriger la valeur et la renvoyer par le biais d’un indicateur dans la face-avant.

Un autre moyen d’associer une commande et un indicateur consiste à utiliser une variable locale. Une variable locale se comporte comme un terminal multiple vis-à-vis d’une commande ou d’un indicateur. Vous pouvez aussi bien lire une variable locale qu’y écrire. Pour voir quelques exemples d’utilisation de ces variables locales, reportez-vous au répertoire `examples\general\locals.llb`. Pour plus d’informations sur les variables locales, veuillez vous reporter à la section intitulée *Les variables locales* du chapitre 22, *Les variables globales et locales*, du *Manuel de l’utilisateur LabVIEW*.

La mise en ouvre de l’éditeur de commandes

Vous pouvez enregistrer une commande personnalisée comme *définition de type* ou *définition de type stricte*. L'éditeur de commandes est particulièrement utile pour les applications volumineuses, puisqu'il vous permet de modifier une commande enregistrée dans un autre VI et d'appliquer ces changements à tous les VIs qui l'utilisent. Pour plus d'informations sur les définitions de type, veuillez vous reporter à la section *Les définitions de type* du chapitre 23, *Les commandes personnalisées et les définitions de types*, du *Manuel de l'utilisateur LabVIEW*.

L’acquisition de données et le contrôle d’instruments



Vous allez apprendre :

- A acquérir des données au moyen de cartes d’acquisition (***Windows, Macintosh et Sun***).
- Ce que sont les fonctions VISA.
- Ce que sont les fonctions GPIB.
- A utiliser le port de communication série de votre ordinateur.
- A utiliser un port série pour communiquer avec d’autres ports série.
- A utiliser le VXI pour l’acquisition de données ***(Windows, Macintosh et Sun).***
- Ce qu’est un driver d’instrument.
- A utiliser un VI de test de réponse en fréquence.

Mise en œuvre de LabVIEW pour acquérir des données

L’une des caractéristiques les plus intéressantes de LabVIEW réside dans sa capacité à acquérir des données à partir de toutes sortes d’équipements. C’est pourquoi, LabVIEW dispose de VIs pour le contrôle des cartes et instruments suivants :

- Cartes d’acquisition de données ***(Windows, Macintosh et Sun)***
- Instruments GPIB (IEEE 488)
- Instruments à port de communication série
- Instruments VXI ***(Windows, Macintosh et Sun)***

Ces VIs utilisent les logiciels drivers de National Instruments, devenus depuis des standards de facto, pour vous offrir un contrôle complet de vos acquisitions de données et du matériel utilisé pour gérer votre système d’instrumentation.

Ce tutorial se limite aux caractéristiques et fonctions de base de LabVIEW. Pour en savoir davantage sur les possibilités d'analyse de LabVIEW, veuillez vous reporter au chapitre 1 intitulé *Introduction to Analysis in LabVIEW*, du *LabVIEW Analysis VI Reference Manual*.

Les cartes d'acquisition de données (Windows, Macintosh et Sun)

National Instruments fabrique tous les composants nécessaires à la construction des systèmes d'acquisition de données. Les cartes enfichables sont disponibles pour tous les environnements, IBM PC/AT, EISA, IBM PS/2 MicroChannel, Macintosh NuBus Series, Macintosh LC/LCII, et les ordinateurs SPARCstation SBus.

Toutes ces cartes offrent un large éventail de combinaisons d'entrées et de sorties analogiques, numériques et d'horloge. Vous pouvez utiliser en amont des modules SCXI multiplexeurs pour le conditionnement des signaux, ce qui vous permet d'augmenter le nombre de voies d'entrées analogiques pour un prix avantageux. Une grande variété de modules de conditionnement de signaux pour thermocouples, sondes de température à résistance (RTD), entrées en tension et en courant, entrées/sorties numériques de haute puissance, complètent la gamme des produits d'acquisition.

Les VIs d'acquisition de données **DAQ** de LabVIEW permettent de contrôler l'ensemble des matériels de National Instruments. Pour plus d'informations sur la bibliothèque DAQ de LabVIEW, veuillez vous reporter au chapitre 2 intitulé *Installing and Configuring Your Data Acquisition Hardware*, du *LabVIEW Data Acquisition Basics Manual*, qui aborde plus particulièrement la procédure d'installation et de configuration de votre système avec LabVIEW. Cet ouvrage, *LabVIEW Data Acquisition Basics Manual*, comprend également des renseignements qui vous aideront à démarrer la construction d'un système d'acquisition de données avec LabVIEW. Pour obtenir un descriptif détaillé des cartes DAQ par type de matériel, veuillez vous reporter à l'annexe B, *Hardware Capabilities in LabVIEW*, du *LabVIEW Data Acquisition VI Reference Manual*.

Pour consulter des exemples de VIs d'acquisition de données, veuillez vous reporter au répertoire `examples\daq`.

Le contrôle d'instrumentation VISA

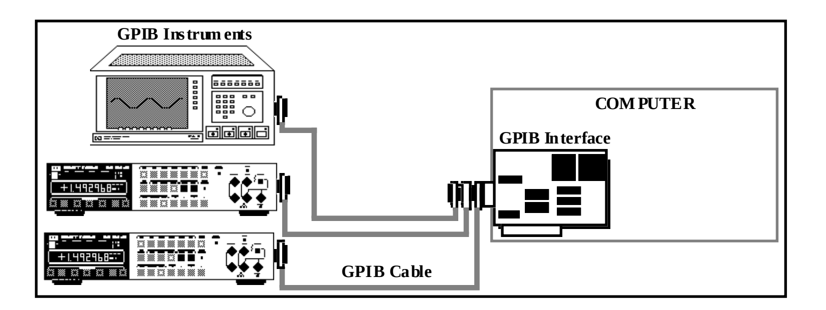
VISA est une bibliothèque d'interfaces qui permet de contrôler à la fois des instruments GPIB, VXI et d'autres types. Grâce aux fonctions VISA, vous pouvez construire un VI de driver d'instrument, qui contrôle à lui seul un modèle particulier d'instrument au moyen de différents supports d'E/S. Une chaîne de caractères est transmise à la fonction **Open** de VISA afin de choisir le type d'E/S à utiliser pour communiquer avec l'instrument en question. Une fois que la session avec l'instrument est ouverte, les fonctions VISA, telles que **VISA Read** et **VISA Write**, exécutent les activités d'E/S de l'instrument de manière générique. Ce qui signifie donc que le programme n'est pas lié aux fonctions GPIB ou VXI. Le driver d'instrument VISA se veut autonome et indépendant de l'interface et peut servir dans plusieurs systèmes différents.

Les drivers d'instrument qui utilisent les fonctions VISA s'articulent autour des activités propres à l'instrument, et non pas au support de communication. Cette particularité offre de nombreuses possibilités quant à l'utilisation d'un driver d'instrument avec un certain nombre de programmes.

Pour consulter des exemples de fonctions VISA, reportez-vous au répertoire `examples\instr`.

Le contrôle d'instrumentation GPIB

Le bus GPIB (*General Purpose Interface Bus*), également appelé IEEE 488, permet de communiquer avec des instruments autonomes comme les multimètres ou les oscilloscopes. National Instruments fabrique de nombreux produits pour le contrôle d'appareils via le bus GPIB. La méthode la plus simple consiste à installer une carte d'interface GPIB dans votre ordinateur et d'y connecter vos instruments au moyen d'un câble GPIB.



Les fonctions GPIB de LabVIEW contrôlent les interfaces GPIB de National Instruments. Vous trouverez la description et l’historique du GPIB, ou de l’IEEE 488, dans la section intitulée *GPIB Overview* du chapitre 1, *Introduction,* dans le *LabVIEW Instrument I/O VI Reference Manual*. LabVIEW utilise le logiciel standard NI-488.2 de National Instruments, livré avec votre interface GPIB.

La bibliothèque GPIB (**Functions»Instrument I/O)** contient à la fois des fonctions traditionnelles GPIB et 488.2. Les fonctions GPIB 488.2 apportent à LabVIEW la compatibilité IEEE 488.2. Ces fonctions contiennent des fonctions spécifiées par la norme IEEE 488.2 et ressemblent aux routines du logiciel NI-488.2 de National Instruments.

Pour consulter des exemples de fonctions GPIB, veuillez vous reporter au répertoire `examples\instr`.

☞ **Remarque :** ***dans la mesure du possible, préférez la fonction VISA à un VI GPIB, compte tenu de sa capacité d’adaptation.***

Les ports série

La liaison série est un moyen de communication très répandu pour la transmission des données entre un ordinateur et un périphérique comme une imprimante, un traceur ou un instrument programmable. La liaison série utilise un émetteur pour envoyer les données, bit par bit, sur une ligne de communication unique à destination d’un récepteur. Cette méthode de communication sert habituellement aux transferts de données à faible vitesse ou sur de longues distances. Les données peuvent ainsi être transférées via des modems sur des lignes téléphoniques traditionnelles.

Le succès de la liaison série s’explique du fait que la plupart des ordinateurs sont dotés d’un ou de deux ports série. Il convient de rappeler toutefois, qu’un port série ne peut communiquer qu’avec un seul périphérique à la fois, d’où ses limites. Pour pouvoir adapter à votre ordinateur plusieurs périphériques, il vous faut utiliser une carte dotée de plusieurs ports série ou d’un module de multiplexage du port série.

LabVIEW dispose de VIs pour le port série, repris en détail au chapitre 8, *Serial Port VIs* du *LabVIEW Instrument I/O VI Reference Manual*. Avant de commencer à utiliser LabVIEW pour les communications série, nous vous conseillons de vérifier d’abord que l’instrument est correctement raccordé à l’ordinateur. Sous Windows, pensez également à contrôler les problèmes d’interruption. L’un des moyens d’y parvenir (Windows et Macintosh) consiste à utiliser un logiciel de gestion de terminaux standard tel que `Terminal` ou `ZTerm` de Microsoft Windows. Une fois la liaison établie avec l’instrument, vous pouvez alors utiliser les VIs de liaison série de LabVIEW qui se trouvent dans la palette **Functions»Instrument I/O»Serial**.

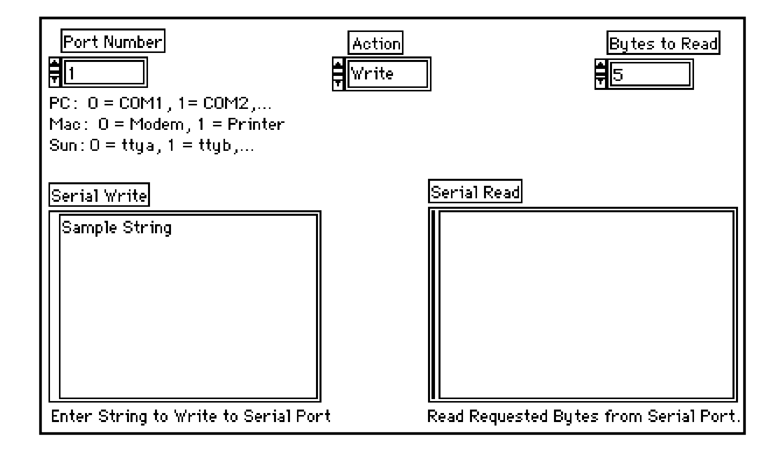
LabVIEW propose cinq VIs pour les communications série : **Serial Port Init**, **Serial Port Write**, **Serial Port Read**, **Bytes at Serial Port** et **Serial Port Break**. Ces fonctions sont reprises en détail au chapitre 8 intitulé, *Serial Port VIs*, du *LabVIEW Instrument I/O VI Reference Manual*.

Pour consulter des exemples de VIs port série, veuillez vous reporter au répertoire `examples\instr\smplserl.llb`.

La mise en œuvre des ports série

OBJECTIF Examiner un exemple de port série que vous pourrez utiliser pour communiquer avec n'importe quel appareil à liaison série. Vous remarquerez que la communication série se rapproche beaucoup de la communication GPIB, à savoir qu'elle se contente souvent d'écrire et de lire des chaînes de caractères ASCII à destination et en provenance d'un appareil.

La face-avant



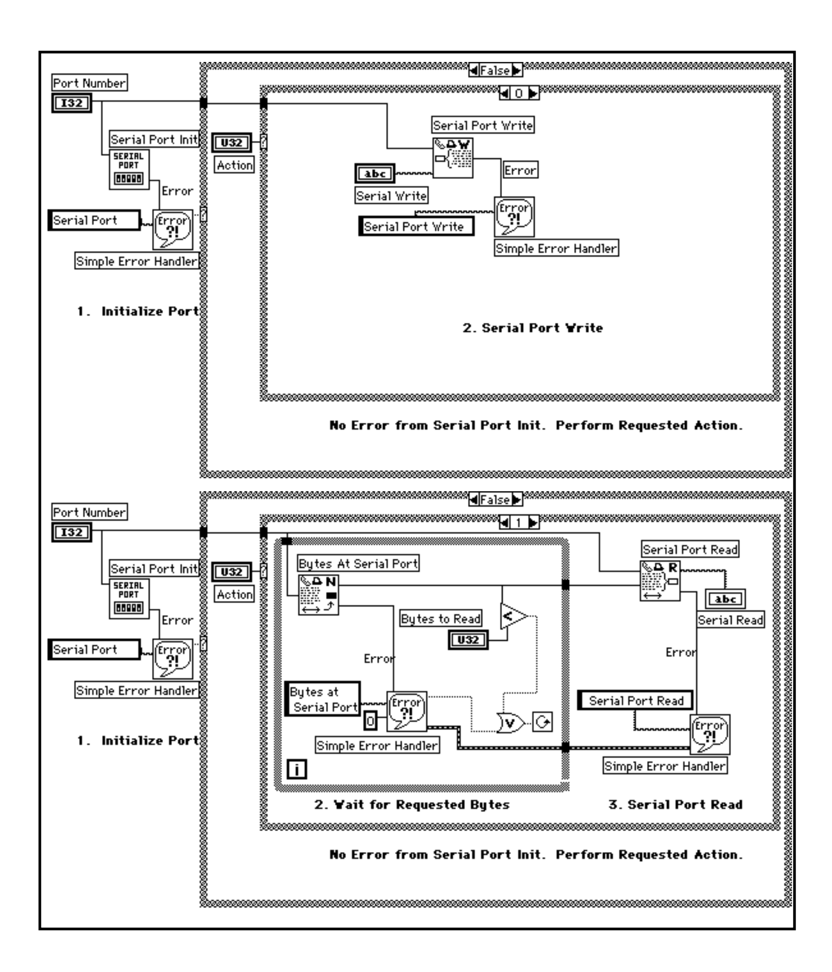
Ouvrez `LabVIEW<->Serial.vi` dans le chemin `examples\instr\smplserl.llb.`

La procédure traditionnelle à suivre dans LabVIEW pour les liaisons série commence avec le VI **Serial Port Init**, figurant à gauche. Ce VI sert à paramétrer le numéro du port série à utiliser, la vitesse de transmission en baud, le nombre de bits de données et de bits stop, et d'autres paramètres nécessaires à une communication série. Ensuite, vous utilisez le VI **Serial Port Write** pour envoyer les commandes nécessaires à l'instrument afin qu'il puisse exécuter l'opération voulue.

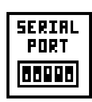
Pour lire des données en provenance d'un instrument série, commencez par lancer le VI **Bytes at Serial Port**. Ce VI sert à vérifier le nombre d'octets en attente au port série. En général, vous utilisez ce VI dans une boucle pour attendre qu'un certain nombre d'octets

s'accumule dans la mémoire tampon avant d'aller lire les informations reçues. Une fois ce nombre atteint, vous utilisez le VI **Serial Port Read** pour passer à l'acquisition de données proprement dite.

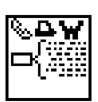
Le diagramme



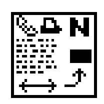
1. Ouvrez le diagramme du VI **LabVIEW<->Serial**, reproduit en partie dans l'illustration ci-dessus, et examinez les fonctions mises en œuvre pour communiquer via le port série conformément à la méthode décrite précédemment. Cliquez sur les flèches situées en haut des structures pour étudier le diagramme dans son intégralité.



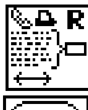
La fonction **Serial Port Init** (**Functions»Instrument I/O»Serial**) initialise le port série sélectionné suivant le paramétrage retenu pour la vitesse de transmission, la taille de la mémoire tampon, le nombre de bits de données, le nombre de bits stop et la parité.



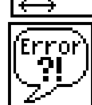
La fonction **Serial Port Write** (**Functions»Instrument I/O»Serial**) écrit une chaîne de données à destination du port série choisi. Une chaîne de caractères de type commande série se compose en général d'un groupe de caractères ASCII.



La fonction **Bytes at Serial Port** (**Functions»Instrument I/O»Serial**) renvoie le nombre d'octets dans la mémoire tampon d'entrée du port série. Vous remarquerez que ce VI reste dans une boucle jusqu'à ce que le nombre requis d'octets soit atteint dans la mémoire tampon. Vous pouvez y ajouter une limitation de temps d'exécution dans le cas où l'appareil n'enverrait pas le nombre d'octets nécessaires.



La fonction **Serial Port Read** (**Functions»Instrument I/O»Serial**) lit le nombre de caractères donné par le port série.



La fonction **Simple Error Handler** (**Functions»Time & Dialog**) signale à l'utilisateur les erreurs éventuelles, les décrit et les localise. Pour plus d'informations sur la gestion des erreurs, veuillez vous reporter à la section intitulée *Quelques informations supplémentaires* à la fin de ce chapitre.

Le contrôle d'instrumentation VXI pour Windows, Macintosh et Sun

Le bus VXI est une plate-forme qui se développe très rapidement, pour les systèmes d'instrumentation. Il utilise un châssis offrant une capacité maximale de treize emplacements capables de recevoir des instruments modulaires sur des cartes enfichables. Les instruments et les châssis sont disponibles en plusieurs tailles chez tous les fabricants. Vous pouvez par ailleurs utiliser des instruments de tailles différentes dans un même châssis. Plusieurs méthodes différentes sont à votre disposition pour contrôler un châssis VXI.

LabVIEW dispose de VIs pour le contrôle de haut et de bas niveau des systèmes VXI. Vous pouvez accéder à ces VIs à partir de **Functions»Instrument I/O»VISA**. Pour plus d'informations sur la manière d'acquérir des données et sur le contrôle des instruments dans un système VXI, veuillez vous reporter au *LabVIEW Instrument I/O VI Reference Manual*.

Les drivers d'instrument

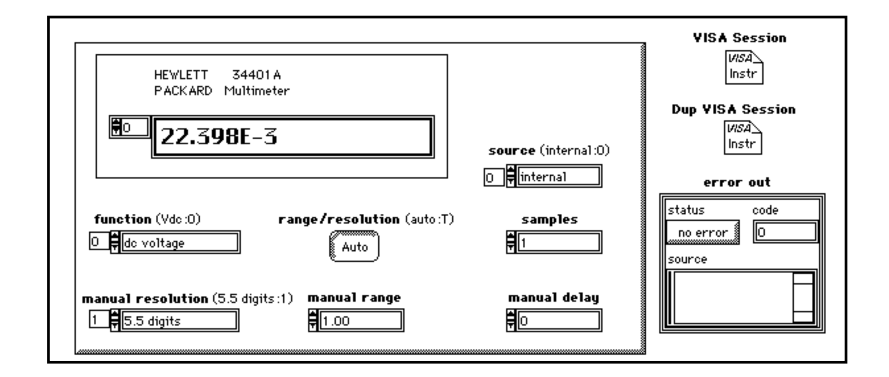
Un driver d'instrument est un logiciel qui permet de contrôler un instrument en particulier. LabVIEW convient tout particulièrement à la création de drivers d'instrument. En effet, la face-avant de LabVIEW simule le fonctionnement d'une véritable face-avant d'instrument. Le diagramme, quant à lui, envoie les commandes nécessaires pour que l'instrument effectue les opérations spécifiées dans la face-avant. Lorsque vous avez terminé la construction d'un driver d'instrument, inutile de retenir les commandes de contrôle de cet instrument. Il vous suffit de définir les valeurs de réglage sur la face-avant. Non pas que cette face-avant logicielle soit d'un grand intérêt en soi pour le contrôle de l'instrument. Là où elle devient intéressante, c'est lorsque vous utilisez le driver d'instrument en tant que sous-VI avec d'autres sous-VIs dans un VI de niveau supérieur pour contrôler un système complet.

LabVIEW dispose d'une bibliothèque composée de plus de 500 drivers d'instrument pour les appareils à interface GPIB, série, CAMAC et VXI (pour Windows, Macintosh et Sun). Compte tenu de la variété des types d'instruments, il nous est impossible de présenter les techniques de création de drivers pour tous les types d'instruments. Sachez cependant que chaque driver construit une *chaîne de commande* qu'il envoie ensuite à l'instrument pour qu'il exécute l'opération dans la face-avant virtuelle. La chaîne de commande comprend des commandes spécifiques à l'appareil (généralement en ASCII) qui le contrôlent à distance. C'est pourquoi, les drivers d'instrument contiennent davantage de fonctions réservées à la manipulation de chaînes qu'à l'interfaçage spécifique. Pour plus d'informations sur les drivers d'instrument, veuillez vous reporter au chapitre 3 intitulé *Developing a LabVIEW Instrument Driver*, du *LabVIEW Instrument I/O VI Reference Manual*.

La mise en œuvre des drivers d’instrument

OBJECTIF Etudier le driver d’instrument conçu pour le multimètre Hewlett Packard 34401A. Bien que ce driver d’instrument soit à l’origine écrit pour le contrôle GPIB, sachez que la manipulation de chaînes de caractères constitue son rôle essentiel.

La face-avant



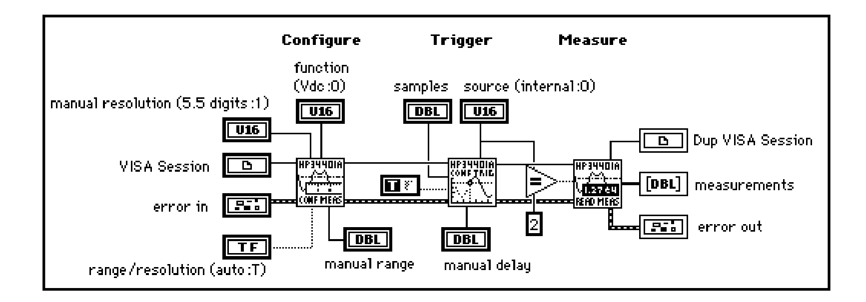
1. Ouvrez le VI `HP34401A Application Example.vi`, qui se trouve dans `labview\examples\instr\hp34401a.llb`.

La face-avant contient plusieurs commandes chargées du paramétrage du multimètre pour différentes mesures.

L’entrée **VISA Session** identifie l’appareil avec lequel le VI communique et transmet toutes les informations liées à la configuration et nécessaires à l’exécution des E/S. Vous devez lancer le VI **HP34401A Initialize** pour établir la liaison avec l’instrument et obtenir la valeur de **VISA Session**.

Vous remarquerez également que le *cluster* **Error Out** décrit toutes les erreurs éventuelles provenant du VI en cours d’exécution.

Le diagramme

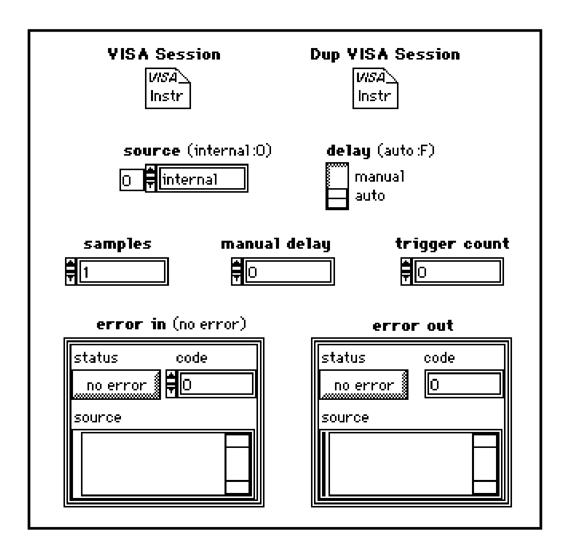


1. Ouvrez le diagramme du VI `HP34401A Application Example.vi`.

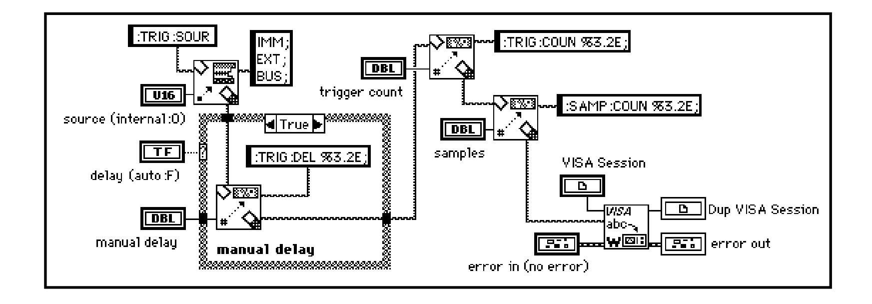
Le diagramme appelle trois sous-VIs : le sous-VI **HP34401A Config Measurement**, le sous-VI **HP34401A Config Trigger** et le sous-VI **HP34401A Read Measurement**.

Les drivers d'instrument de LabVIEW se composent de plusieurs VIs qui permettent de contrôler l'instrument et d'un exemple du mode de fonctionnement de ces VIs. En général, chaque VI exécute une tâche spécifique telle que la configuration, le paramétrage des déclenchements ou bien encore la lecture d'une mesure. Tous les VIs ayant été créés dans LabVIEW, vous pouvez facilement modifier le code de votre application pour combiner plusieurs tâches ou développer certaines fonctionnalités afin d'en augmenter les performances.

2. Ouvrez le sous-VI **HP34401A Config Trigger** en double-cliquant dessus.

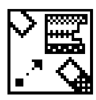


3. Revenez au diagramme et voyez comment le driver d'instrument a été rédigé.



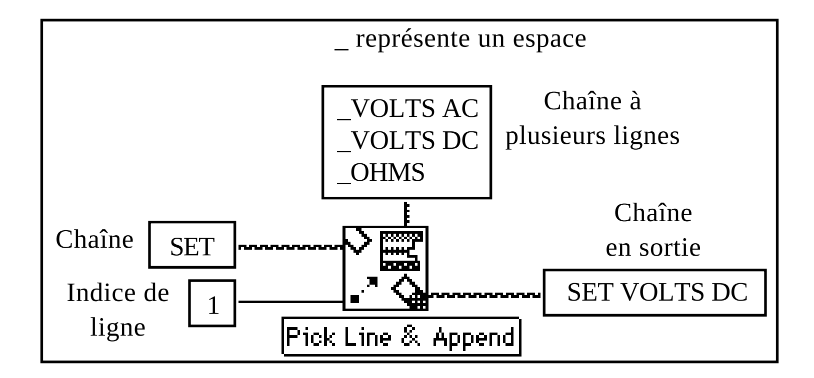
Vous remarquerez que le driver d'instrument n'est rien d'autre qu'un ensemble de fonctions de manipulation de chaînes de caractères pour construire les commandes à adresser à l'instrument. Dans cet exemple, on utilise un VI **VISA Write**, qui envoie la chaîne de commande aux instruments GPIB ou VXI.

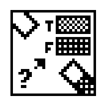
Les fonctions suivantes sont les plus souvent utilisées pour écrire des drivers d'instrument.



La fonction **Pick Line & Append** (**Functions»String**) choisit une ligne particulière d'une chaîne de caractères à plusieurs lignes et l'ajoute à une autre chaîne de caractères.

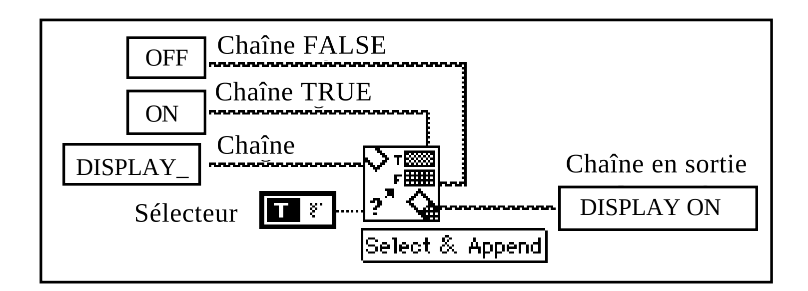
Dans l'exemple qui suit, la fonction choisit la chaîne de caractères `VOLTS DC` et l'ajoute à la chaîne `SET`. Vous remarquerez que l'indice de ligne 1 choisit la deuxième ligne puisque l'indice de la première ligne vaut zéro.

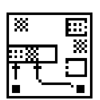




La fonction **Select & Append** (**Functions»String**) choisit une chaîne de caractères en fonction d'un sélecteur booléen et ajoute cette chaîne à la chaîne en sortie.

Dans l'exemple qui suit, le VI ajoute la chaîne (TRUE) `ON` à la chaîne d'entrée `DISPLAY`.





La fonction **Match Pattern** (**Functions»String**) recherche une chaîne de caractères contenant une expression spécifique et renvoie la chaîne correspondante, celle d'avant la recherche, et celle d'après. Cette fonction est une fonction particulièrement puissante. Pour en savoir

davantage, veuillez vous reporter au chapitre 4 intitulé *String Functions* du *LabVIEW Function Reference Manual*.

4. Lorsque vous en aurez terminé avec l'examen du driver d'instrument, fermez le VI sans enregistrer aucune modification.

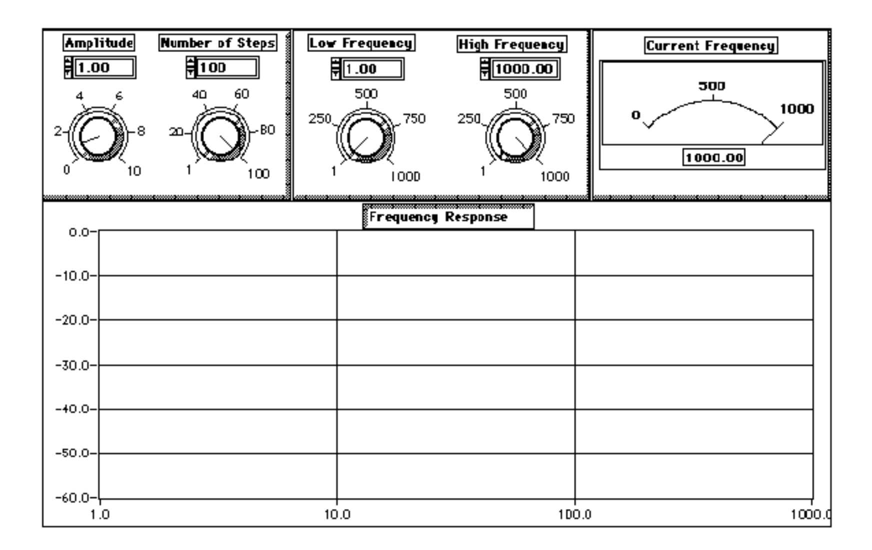
Il existe un grand nombre de drivers d'instrument pour LabVIEW. Si l'instrument que vous utilisez ne dispose pas de VI de driver d'instrument, vous pouvez utiliser un driver d'instrument initialement écrit pour un instrument équivalent ou par le même fabricant et ainsi modifier les chaînes de commande pour l'adapter à votre propre instrument.

La mise en œuvre du VI de test de réponse en fréquence

Généralement, les drivers d'instrument tels que celui que vous avez pu examiner dans l'exercice précédent sont utilisés dans les applications de test et de mesure nécessitant plusieurs instruments. Imaginons une application dans laquelle on utilise des instruments GPIB pour réaliser un test de réponse en fréquence sur un circuit sous test. Un générateur de fonctions délivre une entrée sinusoïdale au circuit sous test et un multimètre numérique mesure la tension en sortie du circuit sous test. Supposons maintenant que vous vouliez examiner la tension obtenue sur un tracé de courbe dans un graphe XY.

OBJECTIF Utiliser un VI qui simule l'utilisation d'instruments GPIB pour réaliser un test de réponse en fréquence sur un circuit sous test présenté au paragraphe précédent.

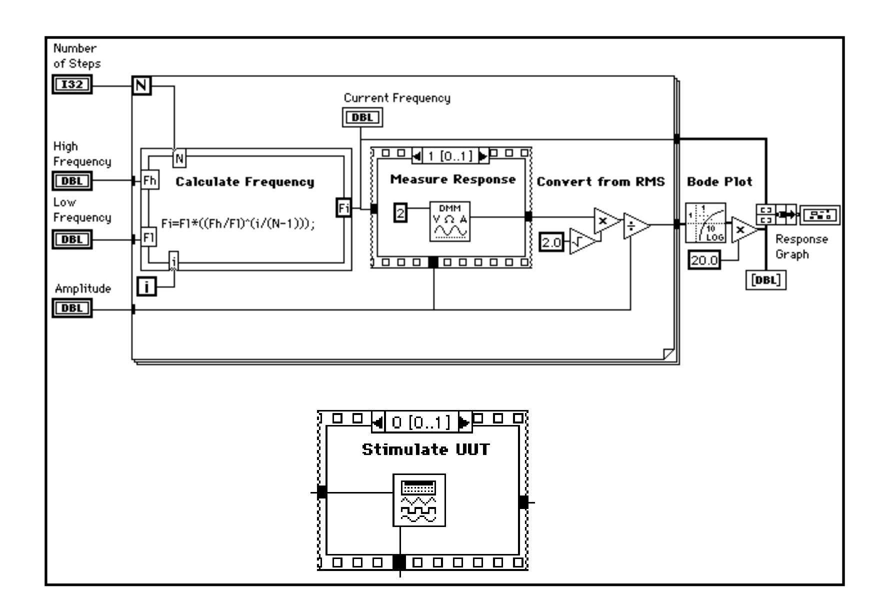
La face-avant



1. Ouvrez `Frequency Response.vi`, dans `examples\apps\freqresp.llb.`

La face-avant contient plusieurs boutons rotatifs permettant de configurer le générateur de fonctions. Vous pouvez ainsi régler l'amplitude, la basse fréquence, la haute fréquence et le nombre de pas de fréquence. Un vumètre sert à afficher la fréquence en cours du générateur de fonctions, et un graphe XY affiche le tracé de la tension obtenue à l'issue du test.

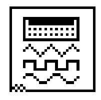
Le diagramme



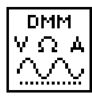
1. Ouvrez le diagramme du VI **Frequency Response**.

Le diagramme contient une boucle *For* au niveau le plus élevé. La boucle s'exécute N fois, comme indiqué par le bouton rotatif `Number Of Steps`.

Une boîte de calcul prend en compte les valeurs indiquées par les boutons rotatifs libellés `Low Frequency`, `High Frequency` et `Number Of Steps` pour calculer la *ième* fréquence de chaque itération en faisant en sorte que les *N* fréquences soient régulièrement espacées sur l'axe logarithmique. Une fois que la boîte de calcul a calculé la *ième* valeur de la fréquence, le nœud la transmet ainsi que celle fournie par le bouton rotatif Amplitude à une structure Séquence.

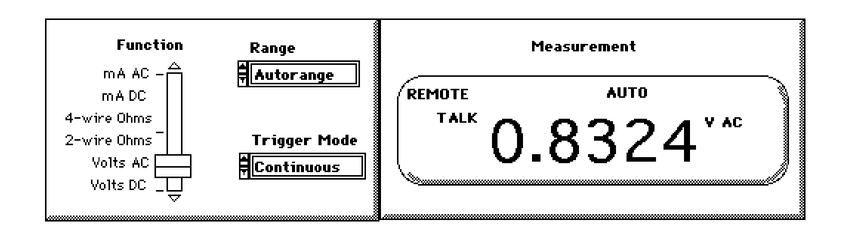


Le cadre 0 contient le sous-VI **Demo Tek FG 5010**. Sa construction ressemble à celle du driver Fluke 45 étudié précédemment. A la différence que le nœud du sous-VI **Demo Tek FG5010** ne nécessite que deux paramètres en entrée, à savoir la fréquence et l'amplitude. La fonction par défaut du driver (sinusoïde) et le mode (signal entretenu) conviennent à cette application et n'ont donc pas besoin d'être modifiés.

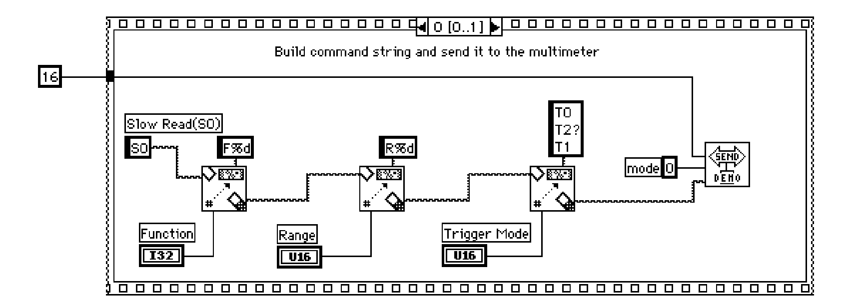


Le cadre 1 contient le sous-VI **Demo Fluke8840A**.

2. Ouvrez le VI **DemoFluke8840A** en double-cliquant dessus à l'aide de l'outil Doigt. Vous remarquerez que la face-avant, reproduite dans l'illustration suivante, est la fidèle reproduction de l'instrument réel.



3. Ouvrez le diagramme, dont on voit une partie dans l'illustration suivante, et voyez comment le driver d'instrument a été rédigé. Vous remarquerez qu'il ressemble étrangement au driver d'instrument étudié précédemment, le multimètre **Fluke 45**.



4. Lorsque vous en aurez terminé avec l'examen du driver d'instrument de démonstration, fermez le VI sans enregistrer aucune modification.

5. Terminez l'observation du diagramme du VI **Frequency Response**.

Le sous-VI **DemoFluke8840A** fournit une tension RMS, qui est ensuite convertie en tension crête-à-crête en la multipliant par la racine carrée de 2. Une courbe de Bode de la réponse en fréquence restitue le gain du circuit sous test en dB par rapport au logarithme de la fréquence multipliée par 0,20 (la tension crête-à-crête est convertie en gain du circuit sous test en dB).

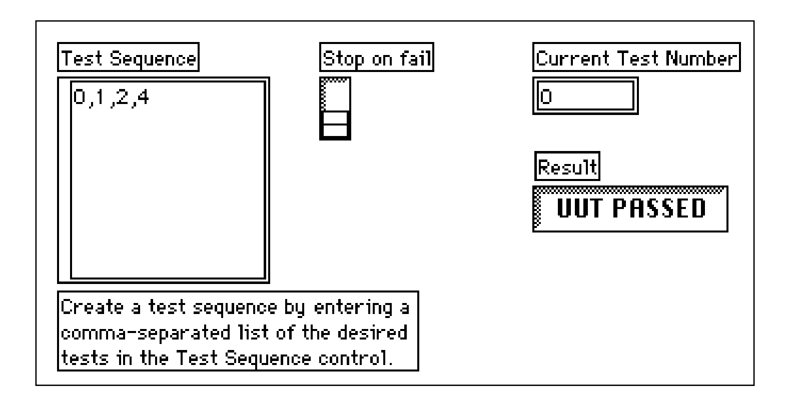
Les tunnels sur chaque bord de la boucle *For* assemblent automatiquement les valeurs de chaque itération dans un tableau en utilisant la caractéristique d'auto-indexation décrite au chapitre 4 intitulé *Tableaux, clusters et graphes* de ce tutorial. Le programme utilise alors une fonction **Bundle** pour exploiter les tableaux en x et en y et les générer sous forme de graphe.

6. Retournez dans la face-avant et lancez le VI. Modifiez le réglage des boutons rotatifs pour pouvoir observer les différences de tracés des réponses en fréquence.
7. Fermez le VI **Frequency Response** sans enregistrer aucune modification éventuelle.

L'écriture d'un séquenceur de test

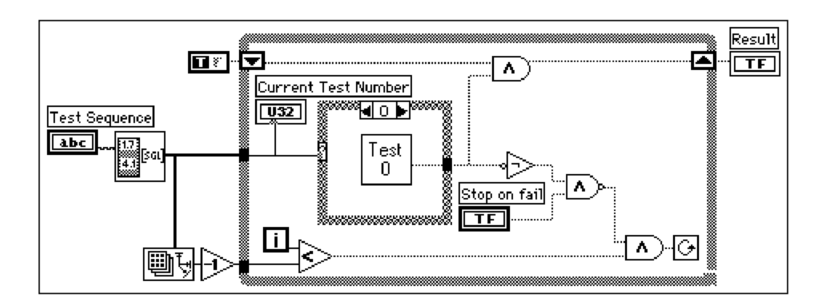
La plupart des applications de production et de test automatique (ATE) à l'instar de celle que vous venez d'étudier, nécessite un séquencement automatique de test. Chaque opération de test mesure une caractéristique particulière du circuit sous test et compare le résultat de la mesure avec une valeur limitée. Si la valeur est inférieure à la limite fixée, le test est réussi. Dans le cas contraire, le test échoue. Une procédure de test complète pour un circuit sous test, se compose d'une série de tests de même nature, exécutés séquentiellement. L'exemple que nous allons maintenant étudier explique comment construire un séquenceur de test.

La face-avant



1. Ouvrez `Test Sequencer.vi` situé dans `examples\apps\testseq.llb.`

Le diagramme



1. Ouvrez le diagramme du **VI Test Sequencer** présenté dans l'illustration précédente.

La séquence de test est une chaîne de caractères que le VI transforme en un tableau numérique de tests à exécuter. Ce tableau est ensuite transféré dans une boucle *While* contenant une structure Condition. Chaque condition contient un VI de test qui correspond à un nombre dans la chaîne de la séquence de test. La boucle *While* parcourt le tableau des numéros de test, pour y sélectionner le test à exécuter. L'indicateur **Current Test Number** permet de savoir quel test est en cours d'exécution. Si un test échoue et que l'opérateur a demandé un arrêt en cas d'erreur, variable sur TRUE, alors la boucle s'arrête.

Dans notre exemple, chaque test transmet une valeur booléenne indiquant si le test a réussi ou non. La variable booléenne vaut TRUE si le test est réussi, et FALSE dans le cas inverse. La boucle du séquenceur de test principal utilise un registre à décalage pour conserver un historique des résultats de tests obtenus (PASS/FAIL) pour la séquence de test dans son intégralité. Lorsque la boucle s'arrête, la séquence de test affiche cet historique dans l'indicateur **Result**.

De nombreuses extensions peuvent venir s'ajouter à ce séquenceur de test. Nous pourrions ainsi imaginer des évolutions permettant de stocker les séquences de test dans des fichiers plutôt que de les présenter dans des commandes de face-avant. L'opérateur peut ainsi

charger la séquence de test qui convient pour le circuit à contrôler. D’autres évolutions classiques servent à :

- Générer le rapport des résultats des tests.
- Inviter l’opérateur à fournir le numéro de série du circuit sous test.
- Reboucler un test non réussi.

Vous pouvez ajouter toutes ces possibilités au séquenceur de test décrit dans cet exercice. Cela étant, vous aurez sans doute besoin du kit logiciel LabVIEW Test Executive Toolkit, qui contient un séquenceur de test entièrement paramétré, toutes les évolutions mentionnées ci-dessus, y compris les utilitaires de tests, les conditions de contrainte, et autres. N’hésitez pas à contacter National Instruments pour toute information concernant l’acquisition de ce kit logiciel.

Résumé

Il existe plusieurs manières d’acquérir des données avec LabVIEW. L’interface GPIB est très utile pour communiquer avec des instruments séparés. La liaison port série sert, quant à elle, à transférer des données sur de longues distances.

(Windows, Macintosh et Sun) Vous pouvez utiliser des VIs pour contrôler des cartes d’acquisition de données ainsi que des systèmes VXI.

(Windows, Macintosh et Sun) Vous pouvez approfondir vos connaissances sur la mise en œuvre des VIs pour acquérir des données au moyen de cartes d’acquisition de données en vous reportant au *LabVIEW Data Acquisition Basics Manual*.

(Toutes les plates-formes) Tout comme les VIs GPIB ne nécessitent aucune connaissance approfondie du bus IEEE 488, les VIs de la bibliothèque des drivers d'instrument de LabVIEW vous évitent de maîtriser parfaitement un instrument particulier. Cette bibliothèque propose plus de 300 drivers d'instrument différents. Si jamais l'instrument que vous utilisez ne figure pas parmi ceux proposés, vous pouvez toujours trouver un instrument très proche et le modifier pour l'adapter à votre propre instrument.

Les fonctions VISA constituent la meilleure méthode pour le contrôle d'instruments, tels que le GPIB, VXI, etc. Etant donné que les fonctions VISA s'articulent uniquement autour des activités propres à

l'instrument, nous pouvons dire que les drivers d'instrument gérés par VISA sont indépendants de l'interface.

Les fonctions GPIB contrôlent les communications GPIB. Les fonctions les plus souvent utilisées sont **GPIB Write**, **GPIB Read** et **GPIB Status**. La fonction **GPIB Write** envoie les données à un instrument. La fonction **GPIB Read** lit les données en provenance d'un instrument. Le VI **GPIB Status** restitue l'état du GPIB au moment d'exécuter le VI. Plusieurs autres fonctions GPIB effectuent des opérations moins utilisées comme l'effacement du paramétrage d'un appareil, l'attente d'une requête de service, le déclenchement d'un appareil ou bien encore l'interrogation d'un appareil.

Vous pouvez contrôler des appareils à interface série avec des VIs de port série. Il n'en existe que cinq. Pour contrôler un appareil à interface série, vous utilisez le VI **Serial Port Init** qui vous sert à configurer le port de communication. Ensuite, vous utilisez le VI **Serial Port Write** pour envoyer une chaîne de commande. Vous pouvez attendre qu'un nombre donné de caractères parvienne à la mémoire tampon avec le VI **Bytes at Serial Port**. Pour finir, vous lisez les données contenues dans la mémoire tampon au moyen du VI **Serial Port Read**.

(Windows, Macintosh et Sun) Vous pouvez contrôler les systèmes VXI si vous disposez du système de développement VXI pour LabVIEW. Pour plus d'informations à ce propos, veuillez vous reporter au *LabVIEW VXI VI Reference Manual*.

LabVIEW dispose de nombreuses fonctions sur les chaînes de caractères qui conviennent parfaitement à la programmation d'instrumentation. Ces fonctions vous aideront à convertir des données d'un type à l'autre ou bien encore à extraire des nombres à partir de chaînes de caractères.

Pendant le processus d'acquisition de données et de contrôle d'instruments, vous pouvez avoir un ensemble d'opérations de test qui contrôlent votre application. Vous pouvez utiliser l'exemple du séquenceur de test pour construire ces opérations de tests. Le kit logiciel LabVIEW Test Executive Toolkit est également disponible si vous souhaitez avoir un contrôle complet de votre application de test automatique ATE. Pour en savoir d'avantage sur ce kit logiciel, n'hésitez pas à contacter National Instruments.

Quelques informations supplémentaires

La gestion d'erreurs

Lorsque vous développez des applications comportant des opérations d'E/S, pensez à utiliser les fonctions de gestion d'erreurs autant que possible. A cet effet, LabVIEW met à votre disposition trois utilitaires de gestion d'erreurs qui sont repris en détail dans le *Manuel de l'utilisateur LabVIEW*. Il s'agit des VIs : **Simple Error Handler**, **Find First Error** et **General Error Handler**. Vous pouvez relier ces trois VIs aux terminaux d'état des erreurs pour détecter les erreurs éventuelles. Le cas échéant, ces VIs donnent une description textuelle de l'erreur. Vous pouvez par ailleurs utiliser ces VIs pour visualiser une boîte de dialogue avec la description du message d'erreur.

Ces VIs de gestion d'erreurs ne contiennent pas uniquement les messages d'erreur de toutes les fonctions GPIB et de port série. Ils affichent également les messages d'erreur pour toutes les E/S sur fichier et les opérations d'analyse.

Les transferts de courbes

La plupart des appareils GPIB qui numérisent l'information, comme les oscilloscopes et les scanners, délivrent des courbes sous forme de chaînes de caractères ASCII ou de nombres binaires. Pour une même courbe, le transfert en binaire s'avère, dans la plupart des cas, plus rapide et moins gourmand en espace mémoire que le transfert d'une chaîne de caractères au format ASCII. Ceci s'explique tout simplement par le fait que le codage binaire nécessite en général beaucoup moins d'octets que le codage en ASCII.

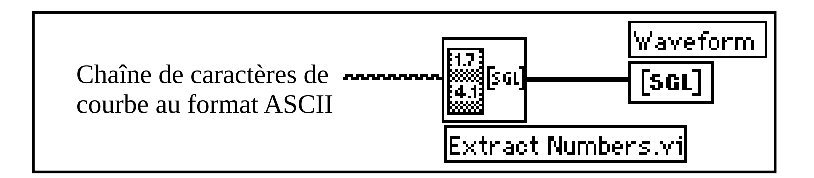
Les courbes ASCII

Imaginons, par exemple, une courbe composée de 1 024 points, chaque point ayant une valeur comprise entre 0 et 255. Pour un codage ASCII, il vous faudrait au maximum 4 octets pour représenter chaque point (3 octets pour la valeur du point, plus 1 octet pour le séparateur, en l'occurrence une virgule). Il vous faudrait donc un maximum de 4 096 (4 * 1 024) octets, plus quelques octets en-tête et quelques autres pour indiquer la fin de la chaîne de caractères. L'illustration suivante reproduit cette courbe sous forme de chaîne de caractères au format ASCII.

```
CURVE {12,28,63,...1024 points in total...,}CR LF
```

En-tête (6 octets)	Point de données (jusqu'à 4 octets chacun)	En-queue (2 octets)

Vous pouvez utiliser le VI **Extract Numbers** (dans le répertoire `examples\general\string.llb`) pour convertir une courbe au format ASCII en un tableau numérique, comme celui représenté ci-dessous.



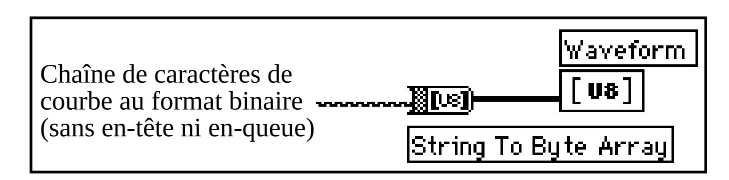
Les courbes binaires

Cette même courbe codée en binaire ne nécessiterait que 1 024 octets (1 * 1 024) plus quelques octets pour l'en-tête et les informations de fin de chaîne. En codage binaire, 1 octet suffit pour représenter le point, en partant du principe que chaque point est un entier non signé sur 8 bits. L'illustration suivante donne un exemple de courbe présentée sous forme de chaîne binaire.

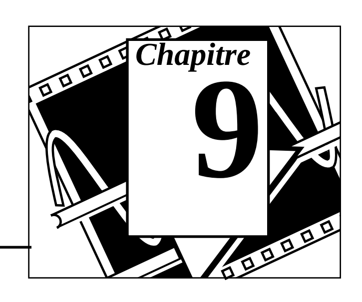
```
CURVE % {MSB}{LSB}{ÂÂ¤å...1024 bytes in total...} {Chk} CR LF
```

En-tête (7 octets)	Compte (4 octets)	Point de données (1 octet chacun)	En-queue (3 octets)

La conversion d'une chaîne binaire en un tableau numérique est un peu plus délicate. Ainsi, il vous faut d'abord convertir la chaîne en un tableau de nombres entiers. Pour ce faire, vous pouvez utiliser la fonction **String To Byte Array** de la palette **Functions»String» Conversion**. Vous devez supprimer tous les octets en-tête et en-queue de la chaîne avant de pouvoir la convertir en un tableau. Faute de quoi, ces informations seraient également converties.



Les techniques et astuces de programmation et de mise au point



Vous allez apprendre :

- Quelques astuces sur le développement des VIs.
- Les techniques de mise au point des VIs.

Quelques astuces de développement

Voici quelques astuces et techniques qui vous aideront dans le développement des VIs dans LabVIEW.

Conseil 1 Les options de menus les plus courantes disposent de leurs propres raccourcis clavier. Par exemple, pour enregistrer un VI, vous pouvez choisir l’option **File»Save**, ou bien appuyer sur la touche <`Ctrl-s`> (Windows), <`command-s`> (Macintosh), <`meta-s`> (Sun) ou <`Alt-s`>.

Action	Windows	Macintosh	Sun	HP-UX
Pour enregistrer un VI	`Ctrl-s`	`command-s`	`meta-s`	`Alt-s`
Pour lancer un VI	`Ctrl-r`	`command-r`	`meta-r`	`Alt-r`
Pour passer de la face-avant au diagramme	`Ctrl-e`	`command-e`	`meta-e`	`Alt-e`
Pour appeler et fermer la fenêtre d’aide	`Ctrl-h`	`command-h`	`meta-h`	`Alt-h`
Pour supprimer toutes les mauvaises connexions	`Ctrl-b`	`command-b`	`meta-b`	`Alt-b`

Action	Windows	Macintosh	Sun	HP-UX
Pour dresser la liste de toutes les erreurs d'un VI	`Ctrl-l`	`command-l`	`meta-l`	`Alt-l`
Pour fermer la fenêtre active	`Ctrl-w`	`command-w`	`meta-w`	`Alt-w`

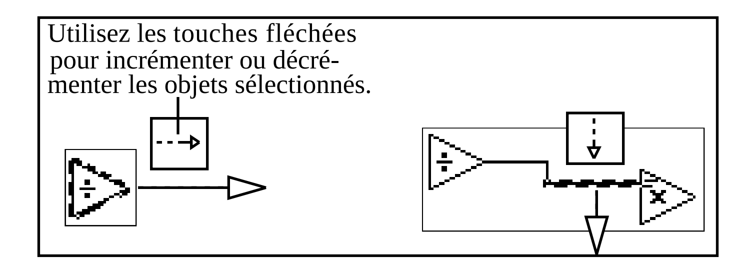
Conseil 2 Pour passer d'un outil à l'autre sur la palette **Tools**, appuyez sur la touche de tabulation <`Tab`>.

Conseil 3 Pour passer de l'outil Doigt à l'outil Flèche dans la face-avant, appuyez sur la barre d'espacement. Dans le diagramme, la barre d'espacement permet de passer de l'outil Flèche à l'outil Bobine.

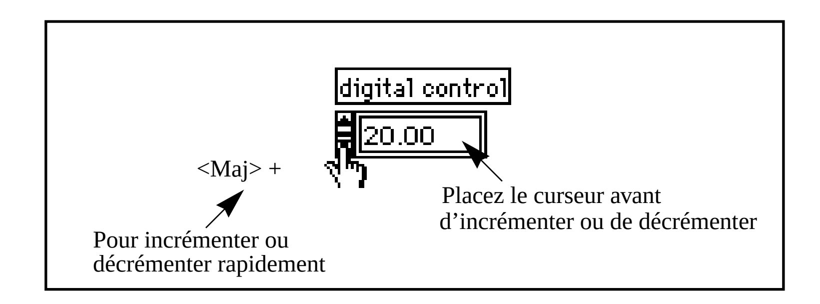
Conseil 4 Pour convertir les palettes **Controls** ou **Functions** ou les sous-palettes en palettes flottantes, appuyez sur la punaise qui se trouve en haut à gauche de la palette.

Conseil 5 Pour modifier la direction d'un fil pendant l'opération de câblage, appuyez sur la barre d'espacement.

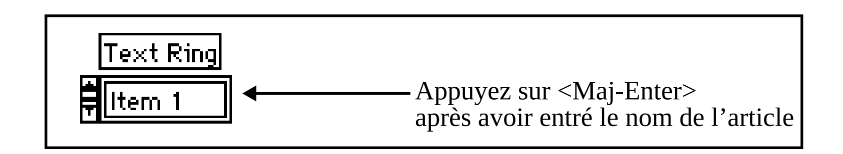
Conseil 6 Pour déplacer un objet sélectionné dans la fenêtre de face-avant ou du diagramme, appuyez sur les touches fléchées du clavier. Ainsi, vous déplacez l'objet sélectionné d'un pixel dans le sens de la flèche. Ce conseil est également valable pour déplacer des portions de fils sélectionnés.

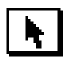


Conseil 7 Pour incrémenter ou décrémenter rapidement, appuyez sur la touche `<Maj>` tout en cliquant sur le bouton d'incrémentation ou de décrémentation d'une commande numérique.

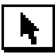


Conseil 8 Pour ajouter des éléments à des commandes de type roue codeuse, appuyez sur `<Maj-Enter>` (Windows) <`Maj-return`> (Macintosh) <`Maj-Return`> (Sun) ou <`Maj-Enter`> (HP-UX) après avoir tapé le nom de l'élément en question. En appuyant sur ces touches, vous validez votre choix et positionnez le curseur pour permettre d'ajouter un autre article.



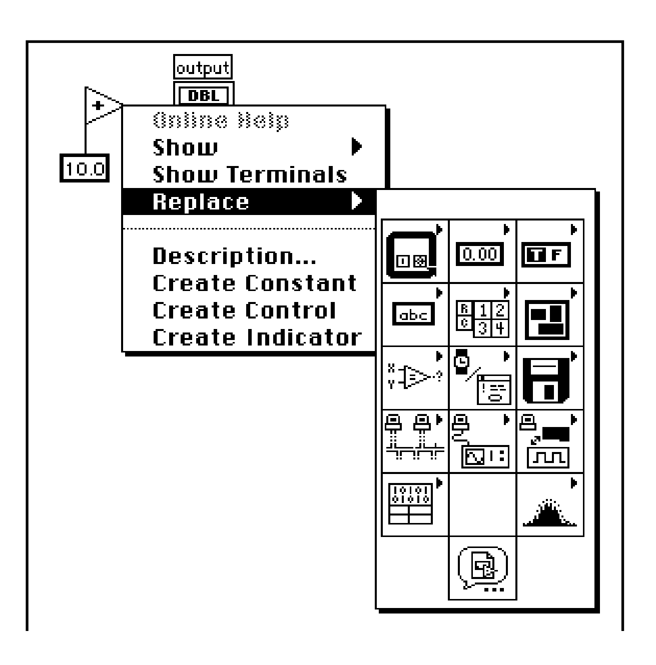


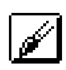
Conseil 9 Pour reproduire un objet, sélectionnez-le à l'aide de l'outil Flèche, maintenez enfoncée la touche `<Ctrl>` (Windows), `<option>` (Macintosh) <`meta`> (Sun) ou <`Alt`> (HP-UX), puis faites glisser la souris.

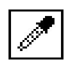


Conseil 10 Pour déplacer un objet dans le sens horizontal ou vertical uniquement, maintenez enfoncée la touche <`Maj`> puis faites glisser l'objet à l'aide de l'outil Flèche.

Conseil 11 Pour remplacer un nœud, ouvrez un menu local sur le nœud en question, puis choisissez l'option **Replace**.







Conseil 12 Pour prélever la couleur d'un objet, commencez par choisir l'outil Pinceau. Placez cet outil sur l'objet choisi, puis appuyez sur la touche <`Ctrl`> (Windows), <`option`> (Macintosh), <`meta`> (Sun) ou <`Alt`> (HP-UX) en la maintenant enfoncée. L'outil se transforme alors en outil Pipette. Prélevez la couleur de l'objet en cliquant dessus. Relâchez la touche et coloriez d'autres objets en cliquant dessus à l'aide de l'outil Pinceau.

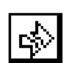
Conseil 13 L'erreur la plus répandue est celle qui consiste à câbler entre elles deux commandes ou à câbler deux commandes à un indicateur. Vous obtenez alors le message d'erreur suivant : `Signal: has multiple sources`. Pour résoudre ce problème, ouvrez un menu local sur la commande puis choisissez l'option **Change to Indicator**.

Conseil 14 Pour créer et câbler automatiquement les bons types de constantes, commandes ou indicateurs à un objet, ouvrez un menu local sur l'entrée ou la sortie de l'objet, puis sélectionnez **Create Constant**, **Create Control** ou **Create Indicator**.

Conseil 15 ***(Windows et UNIX)*** Pour supprimer un câble en cours de connexion, cliquez sur le bouton droit de la souris.

Les techniques de mise au point

La localisation des erreurs



Si votre VI n'est pas exécutable, une flèche *brisée*, représentée à gauche, apparaît sur le bouton Exécution de la barre d'outils. Pour visualiser la liste des erreurs possibles, cliquez sur le bouton Flèche brisée. Cliquez ensuite sur l'une des erreurs de la liste pour mettre en surbrillance l'objet ou le terminal en défaut.



Remarque : ***lorsqu'elle est ouverte, la fenêtre d'affichage des erreurs contient des mises en garde, même si elles ne provoquent pas la cassure de la flèche. (Des nœuds ou des terminaux qui se chevauchent ou qui sont masqués en partie sont des exemples de mises en garde indiquant par là l'existence d'un problème plus profond de conception dans votre VI.)***

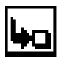
Le mode pas à pas



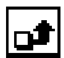
Pour la mise au point, il est parfois préférable d'exécuter un diagramme nœud après nœud. Pour passer en mode Pas à pas, cliquez sur le bouton Exécution semi-détaillée qui se trouve dans la barre d'outils.



Pour exécuter sans détailler une boucle, un sous-VI, etc., cliquez sur le bouton Exécution semi-détaillée de la barre d'outils.

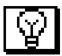


Pour exécuter de façon détaillée une boucle, un sous-VI, etc., cliquez sur le bouton Exécution détaillée de la barre d'outils.



Pour sortir d'une boucle, d'un sous-VI, etc., cliquez sur le bouton Sortie de la barre d'outils. Vous pouvez définir le niveau d'exécution d'un VI et demander à ce que l'exécution s'interrompe à un instant donné en cliquant sur le bouton Sortie tout en maintenant le bouton de la souris enfoncé. Cette manipulation vous permet d'accéder à un menu local.

Le mode Animation



Vous pouvez animer votre diagramme pendant l'exécution d'un VI en cliquant sur le bouton Ampoule de la barre d'outils.



L'ampoule de ce bouton se transforme alors en une ampoule allumée. Vous pouvez utiliser le mode Animation en mode Pas à pas pour suivre le flux des données dans le diagramme.

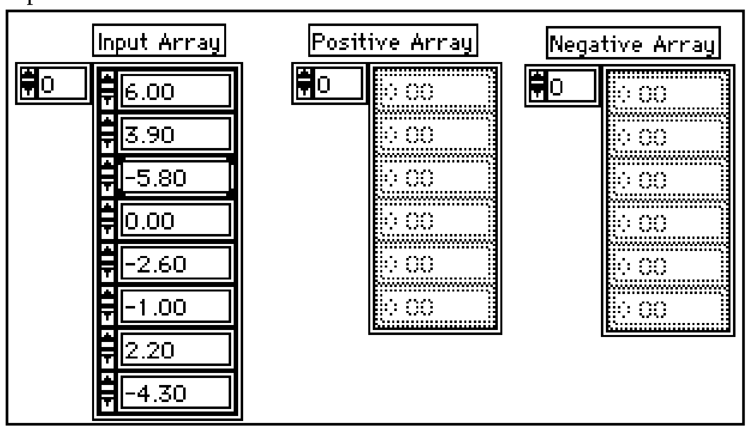
La mise au point d'un VI

OBJECTIF Ouvrir un VI qui va tester tous les nombres d'un tableau pour vérifier s'ils sont négatifs. Le cas échéant, le VI extrait le nombre négatif du tableau pour le mettre dans un autre tableau. En d'autres termes, le VI répartit un tableau d'entrée contenant à la fois des nombres positifs et négatifs dans deux tableaux différents, un tableau réservé aux nombres positifs et un autre aux nombres négatifs.

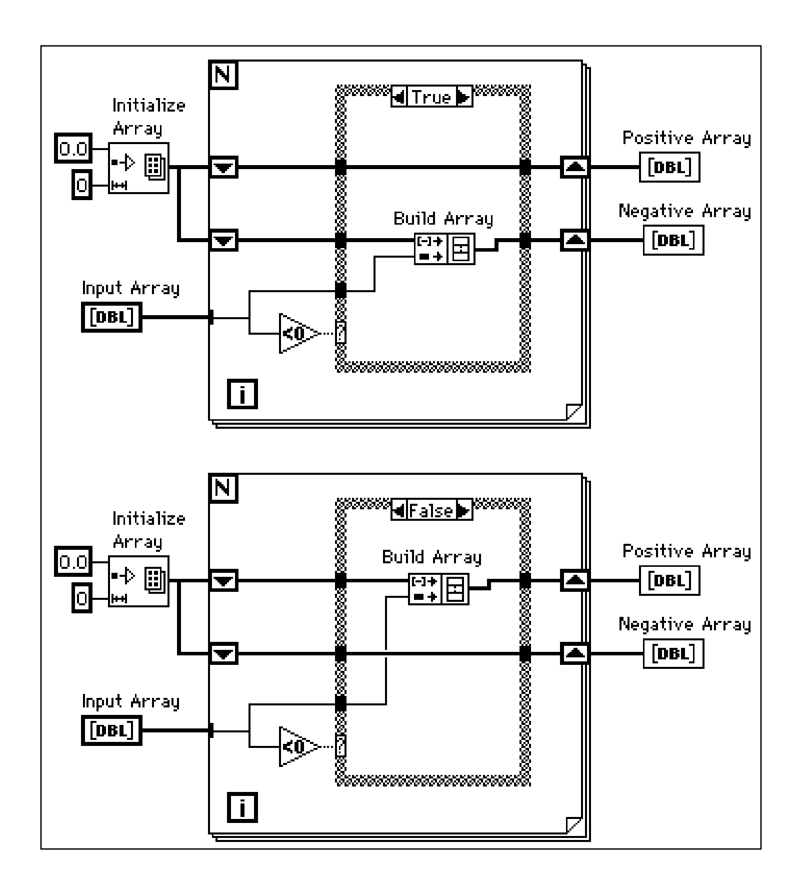
La face-avant

1. Ouvrez `Separate Array Values.vi` en choisissant **File» Open...**. Le VI se trouve dans le répertoire `examples\general\arrays.llb`.

Le tableau de commandes numériques contient les entrées. Les indicateurs **Positive Array** et **Negative Array** affichent les résultats de la séparation.



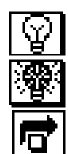
Le diagramme



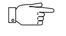
1. Ouvrez et étudiez le diagramme. Vous ne pouvez afficher qu'une seule condition à la fois. Pour passer d'une condition à l'autre, cliquez sur les flèches situées en haut de la structure Condition.

Vous remarquerez que le terminal de comptage de la boucle *For* n'est câblé à aucune valeur particulière, et que le tableau d'entrée est automatiquement indexé au moment où il entre dans la boucle *For*. La boucle *For* se déroulera jusqu'à ce qu'il n'y ait plus aucun élément dans le tableau.

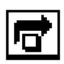
Une excellente façon de mettre au point un VI consiste à l'exécuter en mode Pas à pas et à animer le flux des données qui traverse le diagramme.



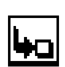
2. Activez le mode Animation en cliquant sur le bouton Ampoule de la barre d'outils du diagramme. Le bouton Ampoule se transforme alors en Ampoule allumée.
3. Activez le mode Pas à pas en cliquant sur le bouton Exécution semi-détaillée de la barre d'outils. Le VI transmet le tableau d'entrée **Input Array** dans la boucle *For*, puis communique les entiers et les nombres à virgule flottante des constantes numériques au tableau d'initialisation **Initialize Array**.



Remarque : ***ces points mobiles représentent le flux des données contenues dans le diagramme. Le nombre d'éléments contenus dans les différents tableaux est répertorié et des valeurs spécifiques s'affichent au fur et à mesure que le diagramme s'exécute.***



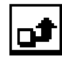
4. Cliquez sur le bouton Exécution semi-détaillée. Vous initialisez ainsi les registres à décalage.



5. Cliquez sur le bouton Exécution détaillée pour exécuter de façon détaillée la boucle *For*. Les valeurs de la fonction **Initialize Array** traversent les registres à décalage puis s'immobilisent dans la structure Condition. La fonction **Less Than** vérifie si la valeur provenant du tableau d'entrée est inférieure à zéro, sachant que pour cette itération la valeur est de six, puis interrompt l'exécution.



6. Cliquez sur le bouton Exécution semi-détaillée pour transmettre le résultat TRUE ou FALSE de la fonction **Less Than** à la structure Condition. Pour cette itération, la valeur d'entrée étant supérieure à zéro, la fonction **Less Than** transmet FALSE à la structure Condition.

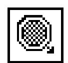


7. Cliquez sur le bouton Sortie. Cette manipulation a pour effet d'exécuter le reste du diagramme ainsi que les itérations suivantes. Vous remarquerez que les éléments ne passent pas encore de la boucle *For* vers les indicateurs. Lorsque toutes les itérations de la boucle sont terminées, le VI transfère les tableaux positifs et négatifs vers les indicateurs. Ce concept est essentiel dans la programmation par flux de données : à savoir qu'une boucle ne s'exécute pas tant que toutes les données qu'elle reçoit ne sont pas disponibles, et que les données ne sortent de la boucle que lorsque son exécution est terminée.

LabVIEW contient également un outil Point d'arrêt qui permet de placer des points d'arrêt sur les nœuds, les diagrammes, les objets des structures et les connexions.



Remarque : ***les cadres rouges qui entourent les nœuds et les diagrammes ainsi que les points rouges sur les connexions indiquent l'emplacement des points d'arrêt.***

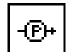


8. Choisissez l'outil Point d'arrêt de la palette **Tools**.
9. Placez le curseur **Point d'arrêt** sur le nœud du tableau d'initialisation **Initialize Array** puis cliquez dessus. Un cadre rouge encercle alors le nœud du tableau.

Remarque : ***assurez-vous que la flèche qui se trouve sur le curseur Point d'arrêt est bien orientée vers l'élément, structure ou câble, sur lequel vous souhaitez fixer ou installer le point d'arrêt.***

10. Parcourez le VI en mode Pas à pas en cliquant sur le bouton Exécution détaillée de la barre d'outils. LabVIEW met en surbrillance le nœud de **Initialize Array** et interrompt l'exécution juste avant que le nœud s'exécute.
11. Cliquez sur le curseur **Point d'arrêt** du nœud **Initialize Array** pour supprimer le point d'arrêt.

LabVIEW contient également un outil Sonde qui permet de visualiser les données pendant qu'elles circulent dans les fils de connexion.



12. Choisissez l'outil Sonde dans la palette **Tools**.
13. Placez une sonde sur le câble reliant le tableau d'entrée **Input Array** à la structure Condition. Une fenêtre libellée **Probe 1** apparaît à l'écran et vous pouvez voir un glyphe jaune avec le chiffre 1 sur le câble. La fenêtre de la sonde apparaît à la fois dans la face-avant et dans le diagramme.
14. Parcourez de nouveau le VI en mode Pas à pas. La fenêtre de la sonde affiche la valeur des données, en l'occurrence 6.0 pour cette itération, lorsqu'elle passe par la portion du fil sélectionnée.



15. Désactivez le mode Animation en cliquant sur le bouton Ampoule.

    Le bouton reprend alors sa forme initiale, symbolisée à gauche.



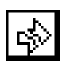
16. Fermez le VI en choisissant **File»Close**. N'enregistrez aucune modification.

L'ouverture des faces-avant des sous-VIs

Une autre technique de mise au point consiste à ouvrir les faces-avant des sous-VIs pour visualiser les données au fur et à mesure qu'elles traversent les sous-VIs. Si par exemple, votre application prévoit un sous-VI pour l'acquisition des données et un autre sous-VI pour l'analyse des données, vous pouvez ouvrir les faces-avant des deux sous-VIs en question, puis lancer l'application principale. Le bouton Exécution du sous-VI se transforme indiquant par là que le sous-VI est en cours d'exécution. Vous pouvez alors vérifier que le sous-VI d'acquisition lit effectivement les bonnes données et que le sous-VI d'analyse les reçoit bien et qu'il calcule les résultats appropriés. Les commandes et les indicateurs des sous-VIs changent lorsqu'ils sont traversés par les données. Vous pouvez utiliser le mode Animation et le mode Pas à pas si vous suspectez un quelconque problème et souhaitez y regarder de plus près.

Si vous souhaitez examiner les sous-VIs d'une application, pensez à ouvrir leurs faces-avant avant de lancer l'application. Sinon, les sous-VIs conservent leurs valeurs par défaut sans que vous puissiez visualiser les valeurs courantes.

Résumé

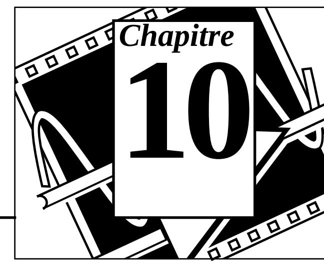


Une flèche brisée dans le bouton Exécution, situé dans la barre d'outils, indique que le VI ne peut s'exécuter. En cliquant sur ce symbole, vous accédez à la boîte de dialogue **Error List** qui affiche la liste des erreurs possibles. Les modes Animation et Pas à pas facilitent la mise au point de vos VIs en vous permettant de suivre le flux des données qui les traversent. L'outil Point d'arrêt vous sera très utile pour la mise au point puisqu'il permet d'interrompre l'exécution d'un VI à votre guise. L'outil Sonde vous permet d'afficher les valeurs acheminées dans un câble pendant l'exécution du VI.

Si vous pensez qu'un sous-VI ne fonctionne pas correctement, ouvrez sa face-avant avant de lancer le VI principal. Vous pourrez ainsi vérifier les valeurs des données qu'il reçoit et qu'il restitue.



La conception des programmes



Félicitations ! Vous venez de terminer ce tutorial et devriez maintenant être familiarisé avec les nombreuses étapes du processus de développement de LabVIEW. Il ne vous reste plus qu’à mettre en pratique toutes ces connaissances en développant vos propres applications.

Comment démarrer ?

Ce chapitre va essayer de répondre à cette question en vous donnant quelques conseils techniques au moment où vous développerez des programmes et en vous donnant par ailleurs quelques astuces pour le style de programmation.

La mise en œuvre de la conception descendante

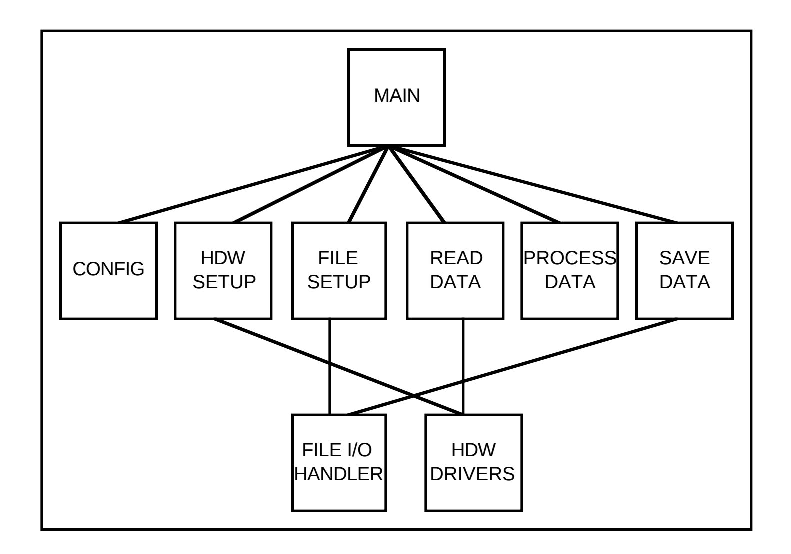
Lorsque vous devez travailler sur un projet volumineux, pensez toujours à la conception *descendante*. C’est ce qui distingue LabVIEW des autres langages de programmation traditionnels. Grâce à la conception descendante, vous pouvez commencer facilement avec l’interface utilisateur graphique puis l’animer.

Dresser la liste du matériel requis

Commencez par dresser la liste des types d’E/S, des vitesses d’échantillonnage, des besoins en analyse temps réel, de la présentation des données etc. Ensuite, créez des faces-avant virtuelles que vous pourrez ensuite présenter aux utilisateurs potentiels (à moins que vous ne soyez l’utilisateur final). Réfléchissez sur les fonctions et leur potentiel. Servez-vous de ce processus interactif pour revoir la conception de l’interface utilisateur si nécessaire. A ce stade-là, il faudra peut-être vérifier que vous êtes bien en mesure de respecter les spécifications, comme le débit de données.

Concevoir la hiérarchie des VIs

Scindez votre travail en plusieurs tâches logiques, faciles à gérer. Comme vous le présente le diagramme suivant, il existe plusieurs blocs principaux que vous rencontrerez sous une forme ou une autre dans tous les systèmes d'acquisition de données.



Vous n'aurez pas toujours besoin de tous ces blocs, ou d'autres blocs seront peut être nécessaires. Par exemple, dans certaines applications, les opérations d'E/S sur fichier sont superflues. Inversement, vous aurez peut être besoin de blocs supplémentaires comme ceux qui illustrent les invites. Quels que soient vos besoins, l'objectif principal est de répartir votre travail de programmation en blocs de niveau supérieur avec lesquels vous pourrez travailler beaucoup plus facilement.

Une fois que vous avez déterminé les blocs dont vous avez besoin, il vous faut tenter de créer un diagramme qui fonctionne avec tous ces blocs principaux. Pour chaque bloc, créez un nouveau VI *à mauvaise connexion* (c'est-à-dire un prototype non fonctionnel qui représente un futur sous-VI). Attribuez-lui une icône, puis créez une face-avant avec les entrées et sorties adéquates. Inutile de créer un diagramme pour ce VI dès à présent. Vérifiez plutôt si ce VI à mauvaise connexion fait bien partie de votre diagramme principal.

Après avoir regroupé plusieurs VIs à mauvaise connexion, essayez de cerner grossièrement la fonction de chaque bloc et la manière dont

chacun de ces blocs fournit les résultats souhaités. Demandez-vous si un bloc donné restitue bien les informations réclamées par le bloc suivant. Le cas échéant, vérifiez que le diagramme principal contient bien les câbles pour transmettre les données entres les différents VIs.

Evitez dans la mesure du possible d'utiliser les *variables globales*. En effet, elles cachent la dépendance des données entre les VIs. Au fur et à mesure que votre système s'étend, il devient de plus en plus difficile à mettre au point si la méthode de transfert des données entre les différents VIs dépend des variables locales.

Ecrire le programme

Vous êtes maintenant fin prêt à écrire le programme dans LabVIEW.

- Pour ce faire, utilisez une approche modulaire en construisant les sous-VIs partout où il existe une répartition logique du travail, ou une réutilisation potentielle du code.
- Résolvez les problèmes plus génériques en même temps que les problèmes plus spécifiques.
- Testez vos sous-VIs à mauvaise connexion au fur et à mesure que vous les écrivez. Vous serez sans doute contraint de construire des sous-programmes de test de niveau supérieur, mais sachez qu'il est beaucoup plus simple de cerner les erreurs dans un petit module que dans une hiérarchie composée de 75 VIs différents.

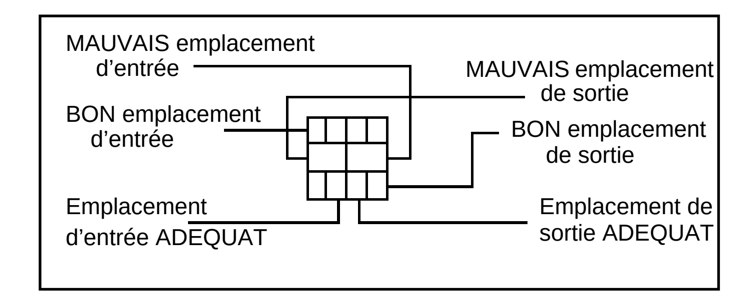
Lorsque vous chercherez à étoffer vos sous-VIs à mauvaise connexion, vous verrez peut-être que votre conception initiale est incomplète. Par exemple, vous vous rendrez compte que vous avez besoin de transférer plus d'informations d'un sous-VI à un autre. Vous devrez alors revoir votre conception principale. C'est à ce titre que l'utilisation de sous-VIs modulaires pour accomplir des tâches spécifiques facilitera la réorganisation de vos programmes.

Planification avec modèles de connecteur

Si vous pensez avoir besoin d'ajouter ultérieurement des entrées et des sorties, choisissez un modèle de connecteur avec des broches supplémentaires. Dans un premier temps, vous pouvez laisser ces broches non connectées. Vous n'aurez ainsi pas besoin de changer de modèle de connecteur pour votre VI si vous estimez avoir besoin d'une autre entrée ou d'une autre sortie par la suite. Sachez que le changement de modèles nécessite le remplacement du sous-VI dans

tous les VIs appelants. Alors qu'en ajoutant des broches supplémentaires, vous pouvez ajouter une entrée ou une sortie sans pour autant trop affecter votre hiérarchie.

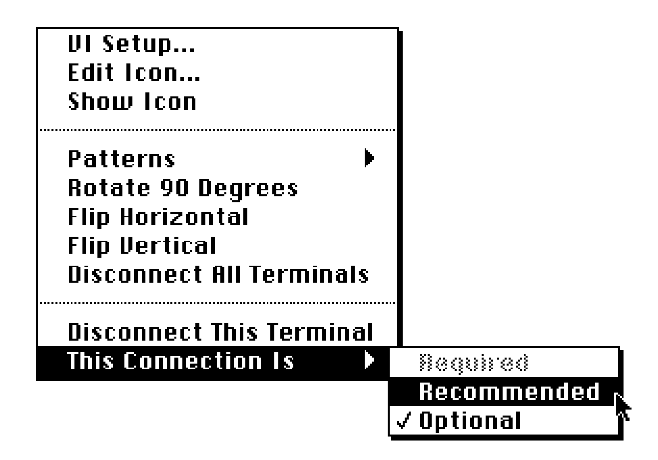
Lorsque vous câblez les commandes et les indicateurs au connecteur, placez les entrées à gauche et les sorties à droite. Vous éviterez ainsi des modèles de liaison compliqués et confus au sein de vos VIs.



Si vous créez un groupe de sous-VIs qui fonctionnent habituellement ensemble, essayez de leur attribuer un modèle de connecteur cohérent, avec des entrées communes au même emplacement. Il vous sera ainsi plus facile de vous souvenir de l'emplacement des entrées sans avoir à recourir à la fenêtre d'aide. Si vous créez un sous-VI dont la sortie sert d'entrée à un autre sous-VI, faites en sorte d'aligner les connexions des entrées et des sorties. Ceci simplifiera vos modèles de câblage.

Les sous-VIs et les entrées nécessaires

Dans la face-avant, vous pouvez éditer les entrées nécessaires aux sous-VIs en cliquant sur le cadre icône situé en haut à droite de la fenêtre, puis en choisissant l'option **Show Connector»This Connection is**. Dans le sous-menu, choisissez entre les options **Required**, **Recommended** ou **Optional**. L'illustration suivante reprend la liste des options de ce sous-menu.



Si vous souhaitez revenir sur le cadre icône de la face-avant, ouvrez un menu local sur le cadre connecteur, puis choisissez l'option **Show Icon**.

Meilleur style de diagramme

Eviter les diagrammes trop volumineux

En général, évitez de créer des diagrammes qui occupent plus d'une ou deux pages écran. Lorsqu'un diagramme s'agrandit, voyez s'il comporte des éléments utilisables dans d'autres VIs, ou bien encore si une partie de votre diagramme peut servir d'élément logique. Le cas échéant, pensez à scinder votre diagramme en plusieurs sous-VIs.

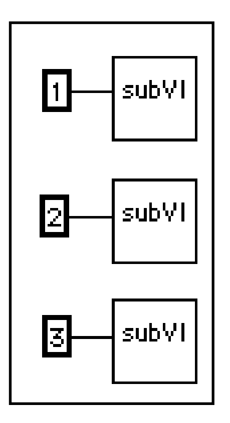
En prévoyant tout avec soin, vous aurez beaucoup moins de mal à concevoir des diagrammes qui utilisent des sous-VIs pour remplir des tâches spécifiques. L'utilisation de sous-VIs vous aide à gérer tout changement et à mettre au point vos diagrammes plus rapidement.

Ainsi, un simple examen suffira à déterminer la fonction d'un programme bien structuré.

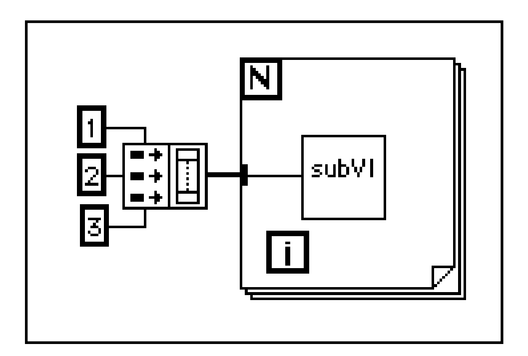
Repérer les opérations courantes

En concevant vos programmes, il vous arrivera de constater que vous répétez souvent une même opération. C'est peut-être l'occasion d'utiliser des sous-VIs ou des boucles pour répéter l'opération en question.

A titre d'exemple, observez le diagramme ci-dessous, où trois opérations identiques sont lancées séparément.



La solution pour remplacer cette conception serait une boucle qui réalise l'opération trois fois de suite. Vous pouvez construire un tableau comportant les différents arguments et utiliser l'auto-indexation pour régler la valeur de chaque itération de la boucle.



Si le tableau contient des éléments constants, vous pouvez utiliser un tableau de constantes plutôt que construire un tableau dans le diagramme.

Dispositions de gauche à droite

LabVIEW a été conçu pour une présentation de gauche à droite, et parfois de haut en bas. Par conséquent, tous les éléments de votre programme devront être disposés de cette manière, dans la mesure du possible.

Vérifier les erreurs

Quel que soit le type d’E/S, ne perdez pas de vue le risque d’erreurs. La plupart des fonctions d’E/S renvoient des informations sur les erreurs rencontrées. Vérifiez que votre programme détecte bien toutes les erreurs et qu’il les traite comme il se doit.

LabVIEW ne traite pas les erreurs automatiquement, car les utilisateurs ont généralement recours à des méthodes de traitement d’erreurs bien spécifiques. Par exemple, si un VI d’E/S dépasse le temps imparti, vous pouvez demander ou non à ce que le programme entier s’interrompe. Peut-être préférez-vous que le VI réessaie encore pendant un certain temps. Dans LabVIEW, c’est à l’utilisateur de prendre les décisions quant à la manière de traiter les erreurs.

La liste suivante reprend trois situations pour lesquelles les erreurs sont fréquentes.

- Mauvaise initialisation des liaisons ou données mal rédigées à l’attention du périphérique externe
- Mauvais fonctionnement du périphérique externe, manque de puissance, ou périphérique défectueux
- Erreurs dans LabVIEW ou dans d’autres programmes qui surviennent lorsque vous mettez à jour LabVIEW ou votre logiciel d’exploitation

Lorsque vous rencontrez une erreur, il est parfois souhaitable d’éviter un certain nombre d’opérations consécutives. Par exemple, si une opération de sorties analogiques échoue parce que vous spécifiez le mauvais périphérique, vous préférerez sans doute qu’une opération de même nature ne se produise pas à nouveau.

Une des méthodes pour traiter ce genre de problèmes consiste à détecter la présence d’erreurs après chaque fonction et à positionner

des fonctions consécutives à l'intérieur des structures Condition. Ceci risque toutefois de compliquer vos diagrammes et de masquer la finalité de votre application.

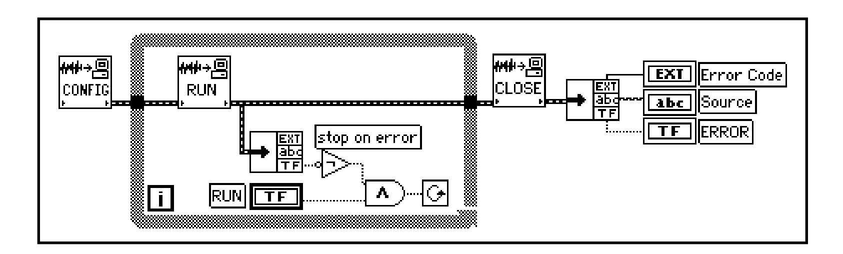
Une autre méthode, efficace dans un grand nombre d'applications et de bibliothèques de VIs, consiste à incorporer le traitement d'erreurs dans les sous-VIs qui gèrent des E/S. Tous les VIs peuvent avoir une entrée ou une sortie d'erreur. Un VI peut être configuré pour détecter toute erreur et savoir si elle est récurrente. En cas d'erreur, le VI peut être configuré pour interrompre l'exécution et transférer l'entrée d'erreur à la sortie d'erreur. Si vous ne rencontrez aucune erreur, le VI peut exécuter l'opération et transmettre le résultat à la sortie d'erreur.

Remarque : ***dans certains cas, comme par exemple dans l'opération* Close*, vous pouvez demander que le VI réalise l'opération sans tenir compte de l'erreur transmise.***

En utilisant la méthode précédente, vous pouvez facilement câbler plusieurs VIs entre eux, connectant ainsi des entrées et sorties d'erreur pour propager les erreurs d'un VI à l'autre. A la fin de séries des VIs, vous pouvez utiliser le VI **Simple Error Handler** pour afficher une boîte de dialogue en cas d'erreur. Le VI **Simple Error Handler** se trouve dans **Functions»Time & Dialog**. En plus d'encapsuler le traitement des erreurs, vous pouvez utiliser cette technique pour préciser l'ordre des opérations d'E/S.

L'utilisation des *clusters* contenant les entrées et les sorties d'erreur présente ce grand avantage qu'ils permettent de contrôler l'ordre d'exécution d'opérations différentes.

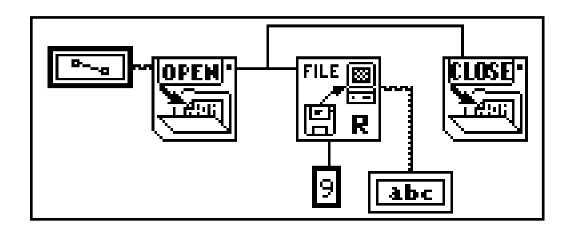
Les informations sur les erreurs figurent en général dans un *cluster* qui contient un code d'erreur numérique, une chaîne de caractères avec le nom de la fonction qui a généré l'erreur, ainsi qu'un booléen de test. L'illustration suivante explique comment réaliser ce type d'opération dans vos propres applications. Vous remarquerez que la boucle *While* s'arrête à chaque fois qu'elle détecte une erreur.



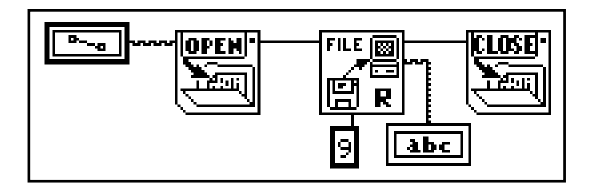
Rechercher des dépendances inexistantes

Assurez-vous que vous avez clairement défini la séquence des événements. Evitez l'exécution descendante ou de gauche à droite lorsqu'il n'existe aucune dépendance entre les données.

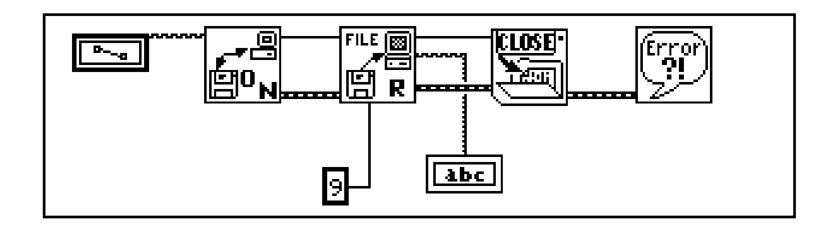
Dans l'exemple suivant, il n'existe aucune dépendance entre **Read File** et **Close File**. Il y a donc de fortes probabilités pour que ce programme ne fonctionne pas comme il devrait.



La version suivante du diagramme établit une dépendance en câblant une sortie de **Read File** sur **Close File**. L'opération ne peut pas s'interrompre avant que **Close File** n'ait reçu la sortie de **Read File**.



Vous remarquerez que l'exemple précédent ne recherche pas d'erreurs éventuelles. Par exemple, si le fichier n'existe pas, le programme ne donne aucune indication dans ce sens. La version suivante du diagramme présente une méthode pour résoudre ce genre de problème. Dans cet exemple, le diagramme utilise les entrées et les sorties d'erreur de ces fonctions pour transmettre toutes les erreurs au VI **Simple Error Handler**.



Eviter de trop utiliser les structures Séquence

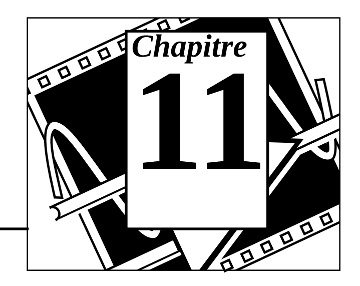
Etant donné que LabVIEW fonctionne sur le principe du parallélisme inhérent, évitez de trop utiliser les structures Séquence. L'utilisation d'une telle structure garantit certainement le bon ordre d'exécution mais empêche aussi le déroulement d'opérations parallèles. Par exemple, des tâches asynchrones qui utilisent des périphériques d'E/S (cartes GPIB, ports série et cartes d'acquisition de données) peuvent fonctionner parallèlement à d'autres opérations si les structures Séquence le permettent.

Par ailleurs, les structures Séquence ont tendance à masquer des parties du programme et à interrompre le flux naturel des données de gauche à droite. L'utilisation de ces structures n'affecte en rien les performances du système. Toutefois, quand vous aurez besoin de séquencer des opérations, vous aurez tout intérêt à envisager l'utilisation du flux de données. Dans les opérations d'E/S, nous vous recommandons d'utiliser la technique d'E/S d'erreurs décrite précédemment pour que les opérations d'E/S se succèdent dans le bon ordre.

Etudier les exemples

Pour en savoir plus sur la conception des programmes, examinez les nombreux exemples de diagrammes fournis dans LabVIEW. Ces exemples de programmes vous donneront un aperçu intéressant sur les styles et la construction des programmes. Pour visualiser ces diagrammes, ouvrez le VI **Readme** qui se trouve dans le répertoire `examples`. Grâce à ce VI, vous pouvez accéder à des exemples variés de programmation.

Développements supplémentaires



Le tutorial que vous venez de parcourir devrait vous aider à créer des applications LabVIEW. Avant de commencer, nous vous recommandons de prendre connaissance des différentes ressources mises à votre disposition.

Vous trouverez dans le répertoire `examples` de nombreux exemples faisant la démonstration des meilleures techniques de programmation. En tête de ce répertoire, se trouve le VI `readme.vi`. Grâce à lui, vous pouvez visualiser les exemples proposés. Lorsque vous choisissez un VI, LabVIEW affiche toute la documentation associée à ce dernier, (informations saisies au préalable dans la boîte de dialogue **VI Information**). Pour ouvrir un VI, choisissez **File»Open...**.

(Windows, Macintosh et Sun) Vous trouverez dans le répertoire d'exemples d'acquisition de données (pour Macintosh, dans le dossier `examples\daq`) une bibliothèque de VIs appelée RUN_ME qui contient un VI d'initiation pour les entrées analogiques, les sorties analogiques, les E/S numériques et les compteurs/timers. Le *LabVIEW Data Acquisition Basics Manual* présente des informations pour chacune des zones fonctionnelles qui vous permettent de parcourir le VI **RUN_ME** et explique comment fonctionnent les VIs d'acquisition de données. Pour plus d'informations sur l'utilisation de ces mêmes VIs avec le matériel SCXI, veuillez vous reporter à la section 5 intitulée *SCXI--Getting Your Signals in Great Condition*, du *LabVIEW Data Acquisition Basics Manual*. Les exemples RUN_ME et le *LabVIEW Data Acquisition Basics Manual* constituent une solide base sur la programmation d'acquisition de données.

Documentation complémentaire

(Windows, Macintosh et Sun) Si vous prévoyez d'intégrer des processus d'acquisition de données dans votre programme, nous vous conseillons de vous reporter au chapitre 3 intitulé *Basic LabVIEW Data Acquisition Concepts*, du *LabVIEW Data Acquisition Basics Manual*. Ce chapitre vous donnera de précieux renseignements quant à l'utilisation des VIs d'acquisition de données avec LabVIEW.

(Toutes les plates-formes) Le *Manuel de l'utilisateur LabVIEW* contient un certain nombre de chapitres qui décrivent les principes de la programmation avancée. Ces principes ne sont pas indispensables dans la plupart des applications. Sachez néanmoins qu'ils existent et qu'ils vous seront très utiles si vous envisagez de développer des applications LabVIEW conséquentes. Ces chapitres abordent les commandes personnalisées ainsi que les définitions de type, la mise au point des performances et les problèmes de portabilité. Vous y trouverez également des informations qui vous aideront à générer des applications et à mieux comprendre comment LabVIEW exécute les VIs. Vous y trouverez par ailleurs de précieux renseignements sur l'impression et la documentation.

Le chapitre 1 du *LabVIEW Communications VI Reference Manual*, aborde les options disponibles pour la gestion de réseaux dans LabVIEW (TCP/IP, DDE, Apple Events et PPC).

Le *LabVIEW Cross Reference Manual* contient un index complet de tous les manuels LabVIEW, un glossaire général ainsi que la liste exhaustive de tous les codes d'erreur.

Informations supplémentaires sur les sujets avancés

L'objectif de ce tutorial est de vous inculquer les principes fondamentaux de la programmation dans LabVIEW. Sachez que certaines fonctions avancées ne sont pas ou très peu abordées. Vous devez malgré tout connaître leur existence pour les mettre en œuvre dans vos applications le moment voulu.

Les *attribute nodes* sont décrits succinctement dans ce tutorial. Ils vous permettent de traiter par programme les paramètres des commandes et des indicateurs. Vous pouvez ainsi modifier la visibilité des commandes, changer par programme les options dans une commande de type roue codeuse, effacer le contenu d'un graphe déroulant ou bien encore modifier les échelles d'un graphe ou d'un graphe déroulant. Les *attribute nodes* sont repris en détail au chapitre 21 intitulé *Les attribute nodes*, du *Manuel de l'utilisateur LabVIEW*.

Ce tutorial a déjà abordé rapidement les variables locales. Vous pouvez les utiliser si vous avez besoin de lire à partir des commandes à plusieurs endroits de votre diagramme. Elles vous seront très utiles pour traiter un objet de face-avant comme une commande dans certains endroits ou un indicateur dans d'autres endroits de façon à ce que vous

puissiez y écrire et y lire. Ces variables locales devront être utilisées à bon escient, dans la mesure où elles masquent le flux de données de vos diagrammes, rendant ainsi plus confuses la finalité de votre programme et la mise au point des variables locales. Veuillez vous reporter au chapitre 22 intitulé *Les variables globales et locales*, du *Manuel de l'utilisateur LabVIEW* pour en savoir davantage sur les variables locales. Vous remarquerez que les applications qui utilisent des variables locales risquent de faire plus de copies de données que celles qui n'en ont pas. Pour en savoir plus sur ce point, veuillez vous reporter au chapitre 27 intitulé *Performances*, du *Manuel de l'utilisateur LabVIEW*.

Vous pouvez utiliser les variables globales si vous avez besoin de stocker les données utilisées par plusieurs VIs différents. A l'instar des variables locales, les variables globales devront être utilisées avec discernement. Ces variables sont indispensables dans certaines applications. Cela étant, il est préférable de ne pas les utiliser si vous pouvez structurer votre programme en utilisant une autre méthode de flux de données pour transférer les données. Pour plus de détails, veuillez vous reporter au chapitre 22 intitulé *Les variables globales et locales*, du *Manuel de l'utilisateur LabVIEW*.

Vous pouvez créer des sous-VIs à partir d'une liste dans le diagramme en utilisant l'option **Edit»Create SubVI from Selection**. En outre, LabVIEW câble automatiquement les bonnes entrées et les bonnes sorties au sous-VI. Dans certains cas, vous ne pouvez pas créer de sous-VI à partir d'un VI. Si vous souhaitez de plus amples informations à ce propos, reportez-vous au chapitre 4 intitulé *La création des sous-VIs*, du *Manuel de l'utilisateur LabVIEW*.

Vous pouvez utiliser la fonction **VI profile** (**Project»Show Profile Window**) pour obtenir des précisions sur les temps de cycle des VIs et leurs statistiques. Cette fonction devrait vous aider à optimiser les performances de vos VIs. Pour en savoir plus sur cette fonction, veuillez vous reporter au chapitre 27 intitulé *Performances*, du *Manuel de l'utilisateur LabVIEW*.

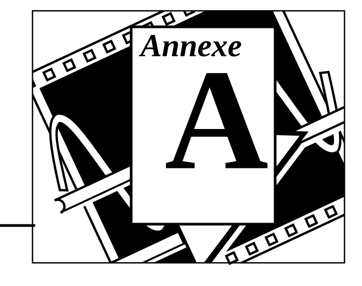
L'éditeur de commandes a été brièvement abordé dans ce tutorial. Vous pouvez l'utiliser pour personnaliser l'aspect de vos commandes. Vous pouvez également vous en servir pour enregistrer des commandes personnalisées afin de les réutiliser dans d'autres applications. Pour une présentation détaillée de l'éditeur de commandes, veuillez vous reporter au chapitre 23 *Les commandes*

personnalisées et les définitions de types, du *Manuel de l'utilisateur LabVIEW*.

Les commandes de type roue codeuse et la liste vous seront très utiles si vous avez besoin de présenter à l'utilisateur la liste des options disponibles. Elles sont abordées au chapitre 14 intitulé *Les commandes et les indicateurs de liste et de type menu rotatif*, du *Manuel de l'utilisateur LabVIEW*.

LabVIEW présente par ailleurs la fonction **Call Library** que vous pouvez utiliser pour appeler une bibliothèque partagée ou une DLL. Avec cette fonction, vous pouvez créer une interface d'appel dans LabVIEW si vous avez besoin d'appeler un code ou un driver existant. Pour plus d'informations à ce sujet, veuillez vous reporter au chapitre 24 intitulé *L'appel de code d'autres langages*, du *Manuel de l'utilisateur LabVIEW*.

Les nœuds d'interface de code (CIN) constituent également une autre méthode pour appeler le code source écrit dans un langage de programmation conventionnel à partir des diagrammes LabVIEW. Ces CIN sont particulièrement utiles pour les tâches que les langages de programmation conventionnels effectuent plus rapidement que LabVIEW, tâches impossibles à effectuer depuis le diagramme, et pour faire le lien entre le code existant et LabVIEW. Cela étant, sachez qu'il est généralement plus facile d'utiliser la fonction d'appel **Library** à partir d'un code source. Il est préférable d'utiliser les CIN pour une intégration plus fine de LabVIEW et du code source. Pour de plus amples informations à ce sujet, reportez-vous au chapitre 24 intitulé *L'appel de code d'autres langages*, du *Manuel de l'utilisateur LabVIEW*.



La parole est à vous

Afin de vous faciliter la tâche, cette annexe prévoit des formulaires types qui vous aideront à nous transmettre des informations. Le premier vous aidera à rassembler les informations dont nous avons besoin pour résoudre les problèmes purement techniques. Le second vous servira à formuler vos remarques sur la documentation fournie avec nos produits. Si vous nous appelez, ayez à portée de main le formulaire destiné au support technique ainsi que le formulaire relatif à votre configuration, si votre manuel le prévoit, afin que nous puissions répondre le plus rapidement possible à vos questions.

National Instruments assure un support technique électronique, téléphonique et par télécopie afin de vous fournir les informations demandées le plus rapidement possible. Sont ainsi à votre disposition différents supports tels que le support BBS, le FTP, le FaxBack ou bien encore le courrier électronique e-mail. Quel que soit le type de problème rencontré, logiciel ou matériel, commencez par utiliser nos systèmes de support technique électronique. Si les informations mises à votre disposition sur ces systèmes ne répondent pas à votre attente, contactez directement par téléphone ou par télécopie notre support technique assuré par des ingénieurs d’application hautement qualifiés.

Services offerts sur support électronique

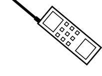
Vous avez la possibilité de vous connecter sur le serveur de fichiers BBS de LabVIEW en utilisant un modem ou le réseau Internet pour bénéficier des services suivants :

- support technique et logiciel
- correspondance électronique
- questions des utilisateurs
- publications techniques, notes d'application
- VIs utilitaires pour applications spécifiques
- mises à jour logicielles et nouveaux drivers d'instrument

Support technique en France

Télécopie

Pour bénéficier du support technique par télécopie, veuillez nous communiquer vos nom et prénom, le nom de votre société, la version de LabVIEW et le type de plate-forme sur lesquelles vous travaillez, ainsi que toutes vos questions les plus détaillées possibles. Composez pour cela le (1) 48 14 24 14.



Téléphone

La possibilité vous est offerte de vous entretenir directement avec nos ingénieurs d'application en composant le : (1) 48 14 24 00.



BBS (modem)

Numéro de téléphone	(1) 48 65 15 59
Débit en bauds	28800
Bits utiles	8
Bits d'arrêt	1
Parité	aucune
Adresse E-mail	`France.support@natinst.com`
Site Internet FTP	
Adresse	`ftp.natinst.com`
Connexion	anonyme
Mot de passe	votre adresse E-mail
Connexion	anonyme
Mot de passe	votre adresse E-mail

Support technique international



FaxBack

Ce système de consultation d'informations automatique contient des fiches sur les produits, les questions les plus souvent posées ainsi que des notes techniques et d'application. Vous pouvez y accéder sur un combiné téléphonique à boutons-poussoirs. Les documents demandés vous sont ensuite télécopiés. Composez le : 19-1 (512) 418-1111 ou le 19-1 (800) 329-7177.

Forums Internet sponsorisés par les utilisateurs

Ces forums permettent aux utilisateurs de communiquer entre eux à propos de LabVIEW.

Demandes d'abonnement : `info-labview-request@pica.army.mil`

Forums : `info-labview@pica.army.mil`

Adresses E-mail

Il s'agit de boîtes à lettres de support technique gérées par les ingénieurs d'application :

```
lv.support@natinst.com
lw.support@natinst.com
hiq.support@natinst.com
gpib.support@natinst.com
daq.support@natinst.com
vxi.support@natinst.com
```

Support téléphonique et par télécopie

National Instruments a des filiales dans le monde entier. Reprenez la liste ci-dessous pour repérer le support technique de votre pays. Si National Instruments n'a pas de filiale dans votre pays, contactez directement le revendeur auprès duquel vous avez acheté votre logiciel pour le support technique.

	Téléphone	Télécopie
Allemagne	089 741 31 30	089 714 60 35
Australie	03 9 879 9422	03 9 879 9179
Autriche	0662 45 79 90 0	0662 45 79 90 19
Belgique	02 757 00 20	02 757 03 11
Canada (Ontario)	519 622 9310	519 622 9311
Canada (Québec)	514 694 8521	514 694 4399
Corée	596 7456	02 596 7455
Danemark	45 76 26 00	45 76 71 11
Espagne	91 640 0085	91 640 0533
Finlande	90 527 2321	90 502 2930
France	1 48 14 24 24	1 48 14 24 14
Hong Kong	2645 3186	2686 8505
Italie	02 48301892	02 48301915
Japon	03 5472 2970	03 5472 2977
Mexique	95 800 010 0793	5 520 3282
Norvège	32 84 84 00	32 84 86 00
Pays-Bas	0348 433466	0348 430673
Royaume-Uni	01635 523545	01635 523154
Singapour	2265886	2265887
Suède	08 730 49 70	08 730 43 70
Suisse	056 200 51 51	056 200 51 55
Taïwan	02 377 1200	02 737 4644

Formulaire pour le support technique

Photocopiez ce formulaire et mettez-le à jour à chaque fois que vous changez votre configuration logicielle ou matérielle. Il vous servira de référence pour votre configuration actuelle. En complétant entièrement ce formulaire avant de nous contacter, vous permettrez à nos ingénieurs d'application d'être plus efficaces pour répondre à vos questions.

Si vous utilisez d'autres produits logiciels ou matériels de National Instruments qui ont un rapport avec le problème en question, reprenez également les formulaires de configuration que vous trouverez dans les manuels d'utilisation de ces produits. Au besoin, prévoyez des pages supplémentaires.

Nom ____

Société ____

Adresse ____

____

Télécopie (___)____________________ Téléphone (___)

____

Marque d'ordinateur _________________ Modèle _________________ Processeur _____________

Système d'exploitation (numéro de version)________________________________

Fréquence d'horloge ______MHz RAM _____Mo Carte graphique _______________________

Souris ___oui ___non Autres adaptateurs installés________________________

Capacité du disque dur _____Mo Marque ______________________________

Instruments utilisés ____________________________________

____

Modèle du produit matériel de National Instruments _____ Révision _______________________

Configuration____

Produit logiciel de National Instruments ___________________________ Version _____________

Configuration____

Problème rencontré : ___________________________________

____

____

____

Liste des messages d'erreurs : ____________________________

____

____

La séquence de manipulations suivante engendre le problème :______________________________

____

____

____

Formulaire pour vos commentaires sur la documentation

National Instruments vous encourage à nous signaler vos remarques sur les documentations fournies avec nos produits. Ces informations nous aideront à vous garantir la qualité que vous attendez de nos produits.

Titre : Tutorial LabVIEW

Date d'édition : Janvier 1996

Numéro de série : 321191A-01

Exhaustivité, clarté et organisation de ce manuel :

____

____

____

____

____

Si vous avez trouvé des erreurs dans ce manuel, indiquez les numéros de pages correspondants et décrivez la nature de ces erreurs.

____

____

____

____

____

____

Nous vous remercions d'avance de votre collaboration.

Nom ____

Fonction ____

Société ____

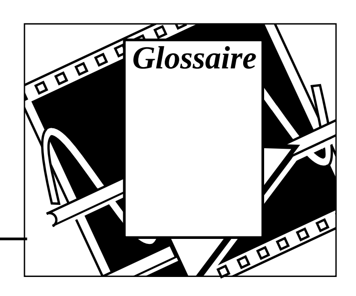
Adresse ____

____

Téléphone (   )____

A envoyer à : National Instruments
Immeuble " Le Continental "
BP 217
93153 Le Blanc-Mesnil Cedex

A faxer à : National Instruments
Immeuble " Le Continental "
(1) 48 14 24 24



Préfixe	Signification	Valeur
m-	milli-	10^{-3}
μ-	micro-	10^{-6}
n-	nano-	10^{-9}

A

absolute path — *Voir* chemin absolu.

acquisition de données — Traitement qui consiste à acquérir des données, généralement au moyen d'une carte de conversion analogique/numérique ou d'une carte enfichable d'entrée numérique.

active window — *Voir* fenêtre active.

ANSI — *American National Standards Institute*. C'est l'organisme américain de normalisation informatique.

ASCII — *American Standard Code for Information Interchange*. C'est un codage normalisé des caractères utilisés par les systèmes informatiques. Il permet de faciliter les transferts de données entre des applications d'origines différentes.

array — *Voir* tableau.

array shell — *Voir* tableau vierge.

auto-indexation — Capacité des structures de boucles à désassembler et à assembler des tableaux à leurs bordures. Lorsqu'un tableau entre dans une boucle dont la fonction d'auto-indexation est activée, il est automatiquement désassemblé, ce qui signifie que les tableaux à une dimension sont transformés en suites de scalaires, les tableaux à deux dimensions en tableaux à une dimension, etc.

	Lorsque les tableaux sortent de la boucle, ils sont réassemblés (processus inverse du précédent).
autoscaling	*Voir* mise à l'échelle automatique.
autosizing	*Voir* dimensionnement automatique.

B

barre de menus	Barre horizontale qui contient les noms des menus principaux.
barre d'outils	Barre contenant des boutons de commande que vous pouvez utiliser pour exécuter et mettre au point les VIs.
bibliothèque de VIs	Fichier particulier qui contient un ensemble de VIs associés.
block diagram	*Voir* diagramme.
boîte de description	Documentation en ligne pour tout objet LabVIEW.
boîte de dialogue	Fenêtre interactive permettant de spécifier des données supplémentaires nécessaires pour compléter une commande.
Boolean controls and indicators	*Voir* commandes et indicateurs booléens.
boucle *For*	Boucle (structure de programmation) itérative qui exécute son sous-diagramme un nombre de fois préétabli. L'équivalent en code traditionnel est une instruction du genre : For i=0 to n-1, do ... .
boucle *While*	Structure de boucle post-itérative qui répète une séquence de code jusqu'à ce qu'une condition soit remplie. Elle est comparable à une boucle DO ou à une boucle REPEAT-UNTIL dans un langage de programmation traditionnel.
breakpoint	*Voir* point d'arrêt.
broken VI	*Voir* VI invalide.
bundle	*Voir* Bundle node.
Bundle node	Fonction qui crée des *clusters* à partir de divers types d'éléments.
byte stream file	*Voir* fichier flux de données.

C

cadre	Sous-diagramme d'une structure Séquence.
cadre connecteur	Zone en haut à droite de la fenêtre de face-avant ou du diagramme, qui représente le terminal d'un VI. Il se trouve sous le cadre icône.
cadre icône	Zone en haut à droite de la fenêtre de face-avant ou du diagramme, qui représente l'icône d'un VI.
caractère non affichable	Caractère ASCII qui n'est pas affichable, tel que le caractère de fin de ligne, la tabulation, etc.
Case structure	*Voir* structure Condition.
chemin absolu	Chemin d'accès à un fichier ou à un répertoire décrivant l'emplacement par rapport au niveau le plus élevé dans l'arborescence du système des fichiers.
CIN	*Code Interface Node*. Nœud particulier d'un diagramme, qui permet de lier un code conventionnel (programmation littérale) à un VI.
cloner	Faire la copie d'une commande ou d'un autre objet de LabVIEW en cliquant sur le bouton de la souris tout en appuyant sur la touche `<Ctrl>` (Windows), `<option>` (Macintosh), `<meta>` (Sun) ou `<Alt>` (HP-UX) et en déplaçant la copie vers son nouvel emplacement. (Sun et HP-UX) Vous pouvez également faire la copie d'un objet en cliquant sur le bouton central de la souris et en déplaçant la copie vers son nouvel emplacement.
cluster	Ensemble ordonné et non indexé d'objets de n'importe quel type : numérique, booléen, chaîne de caractères, tableau ou cluster. Les éléments doivent être soit des commandes soit des indicateurs.
Code Interface Node	*Voir* CIN.
coercition	Conversion automatique qu'exécute LabVIEW pour modifier la représentation numérique d'une donnée.
coercion dot	*Voir* point de coercition.
commande	Objet de la face-avant qui sert à entrer de façon interactive des données à destination d'un VI ou d'un sous-VI par programme.
commande de type roue codeuse	Commande numérique particulière qui associe des entiers à 32 bits démarrant à 0 et augmentant séquentiellement, avec une série d'étiquettes de texte ou de graphiques.

commande et indicateur chaîne de caractères	Objet de la face-avant qui sert à manipuler et à afficher du texte en entrée ou en sortie.
commande graphique	Objet de la face-avant qui permet de représenter des données sur un plan cartésien.
commandes et indicateurs booléens	Objets de la face-avant qui servent à manipuler, afficher ou à traiter en entrée/sortie des données booléennes (TRUE ou FALSE). Il en existe plusieurs types, tels que les interrupteurs, les boutons et les voyants lumineux.
commandes et indicateurs numériques	Objets de la face-avant qui servent à manipuler, afficher, entrer ou sortir des données numériques.
commandes et indicateurs PICT personnalisés	Commandes et indicateurs dont les diverses parties graphiques peuvent être remplacées par vos propres idéogrammes.
compilation	Traitement de conversion du code de haut niveau en code machine. LabVIEW compile automatiquement les VIs avant de les exécuter pour la première fois, après leur création ou toute modification.
condition	Sous-diagramme d'une structure Condition.
conditional terminal	*Voir* terminal conditionnel.
connector pane	*Voir* cadre connecteur.
connecteur	Partie d'un VI ou d'un nœud de fonction qui contient ses terminaux d'entrée et de sortie (à travers lesquels entrent les paramètres et sortent les résultats).
constante	*Voir* constante universelle ou définie par l'utilisateur.
constante définie par l'utilisateur	Objet du diagramme qui émet une valeur définie par l'utilisateur.
constante universelle	Objet non éditable d'un diagramme qui émet un caractère ASCII particulier ou une constante numérique standard, comme pi, par exemple.
control flow	*Voir* programmation séquentielle.
conversion	Changement de type de données.
count terminal	*Voir* terminal de comptage.
CPU	*Central Processing Unit*. C'est l'unité centrale d'un système de traitement (là où sont exécutés les traitements).
curseur	Partie mobile d'une commande ou d'un indicateur coulissant.

custom PICT controls and indicators	*Voir* commandes et indicateurs PICT personnalisés.

D

data logging	*Voir* enregistrement continu de données.
dépendance de données	Condition, dans un langage de programmation par flux de données, qui fait qu'un nœud ne peut s'exécuter que lorsqu'il a reçu des données en provenance d'un ou de plusieurs autres nœuds. *Voir aussi* dépendance de données artificielle.
descripteur de type	*Voir* descripteur de type de données.
descripteur de type de données	Code d'identification des types de données, utilisé pour le stockage et la présentation des données.
diagramme	Description ou représentation graphique d'un programme ou d'un algorithme sous forme de diagrammes. Dans LabVIEW, le diagramme se compose d'icônes exécutables appelées nœuds et de fils qui transportent les données d'un nœud à l'autre. Il s'agit du code source du VI. Le diagramme se trouve dans la fenêtre de diagramme du VI.
dimension	Attributs de taille et de structure d'un tableau.
dimensionnement automatique	Mise à l'échelle automatique des étiquettes pour s'adapter au texte que vous saisissez.
données étalées	Données de n'importe quel type converties en une chaîne de caractères, en général, pour être écrites dans un fichier.
drag	*Voir* glisser.
driver d'instrument	VI qui contrôle un instrument programmable.

E

échelle	Associée à un actionneur mécanique, un graphe déroulant, une commande ou un indicateur graphique, elle contient une série de marques ou de points à intervalles prédéfinis pour repérer des unités de mesure.
éditeur d'icônes	Interface utilisateur identique à celle d'un programme de dessin servant à créer des icônes de VIs.
enregistrement continu de données	Opération généralement mise en œuvre pour acquérir des données et les stocker simultanément dans un fichier sur disque.

	Les fonctions d'entrée/sortie sur fichier de LabVIEW fonctionnent ainsi.
EOF	*End-of-File*. Caractère qui contient la position de fin de fichier par rapport à son début. Il s'agit en fait de la longueur entière du fichier.
E/S (Entrée/Sortie)	Transfert de données de ou vers un système informatique, mettant en œuvre des circuits de communication, des dispositifs de saisie par l'opérateur et/ou des interfaces d'acquisition et de contrôle de données (I/O pour *Input/Output*).
étiquette	Objet recevant du texte, utilisé pour dénommer ou décrire des objets ou des zones dans une face-avant ou un diagramme.
exécution asynchrone	Mode de travail permettant à plusieurs traitements de partager simultanément le même processeur. Un traitement est en cours d'exécution tandis que d'autres attendent, par exemple, une interruption en provenance d'une entrée/sortie ou attendent une impulsion d'horloge.
exécution continue	Mode d'exécution dans lequel un VI s'exécute répétitivement, jusqu'à ce que l'utilisateur l'arrête. On sélectionne ce mode en cliquant sur le bouton Exécution permanente.
exécution “en place”	Possibilité offerte à une fonction ou à un VI de réutiliser une partie de la mémoire au lieu d'en libérer davantage.
execution highlighting	*Voir* mode Animation
exécution pilotée par tableau	Méthode d'exécution par laquelle des tâches individuelles représentent des séquences distinctes au sein d'une structure Condition embarquée à l'intérieur d'une boucle *While*. Ces séquences sont définies par des tableaux de nombres.
exécution réentrante	Mode de travail dans lequel plusieurs appels d'un même sous-VI conduisent à des exécutions en parallèle de celui-ci, chacune disposant de sa propre zone de stockage des données.

F

face-avant	Interface utilisateur interactive d'un VI. Inspirée des faces-avant d'instruments physiques, elle se compose d'interrupteurs, de curseurs, de vumètres, de graphes, de graphes déroulants, de jauges, de voyants lumineux et d'autres commandes et indicateurs.
fenêtre active	Fenêtre dans laquelle l'utilisateur peut saisir des données. C'est généralement la fenêtre la plus en avant sur l'écran. Sur Macintosh, la fenêtre est située dans le Bureau. La barre de titre

	d'une fenêtre active apparaît toujours en surbrillance. Pour activer une fenêtre, il suffit de cliquer dessus ou de la sélectionner dans le menu Windows.
fenêtre d'aide	Fenêtre particulière qui affiche les noms et adresses des terminaux d'une fonction ou d'un sous-VI, la description des commandes et indicateurs, les valeurs des constantes universelles ainsi que la description et le type de données pour les attributs de commande. Cette fenêtre permet également d'accéder à l'aide en ligne LabVIEW.
fenêtre hiérarchie	Fenêtre présentant un affichage graphique des VIs et des sous-VIs.
fichier d'enregistrement de données	Fichier qui stocke les données sous la forme d'une séquence d'enregistrements de type arbitraire et unique, que l'on spécifie à la création du fichier. Même si tous les enregistrements d'un tel fichier doivent être d'un seul type, celui-ci peut être complexe. Par exemple, on peut décider que chaque enregistrement est un *cluster* composé d'une chaîne de caractères, d'un nombre et d'un tableau.
fichier flux de données	Fichier qui contient des données sous forme de séquence de caractères ASCII ou d'octets.
fil	Chemin que suivent les données pour passer d'un nœud à un autre.
file refnum	*Voir* numéro de référence.
flattened data	*Voir* données étalées.
flux de données	Système de programmation composé de nœuds exécutables qui reçoivent et génèrent des données. Les nœuds ne s'exécutent que lorsqu'ils ont reçu toutes les données nécessaires à leur exécution. Ils génèrent ensuite automatiquement des sorties. LabVIEW est un système de programmation par flux de données.
fonction	Elément intégré d'exécution, comparable à un opérateur, à une fonction ou à une instruction dans un langage conventionnel.
formats de stockage	Présentation des données stockées en mémoire.

formula node	Nœud qui exécute des formules mathématiques que vous saisissez comme du texte. C'est particulièrement utile pour les longues formules fastidieuses à construire sous forme de diagrammes.
frame	*Voir* cadre.
free label	*Voir* texte libre.
front panel	*Voir* face-avant.

G

G	Langage de programmation graphique de LabVIEW.
glisser	Déplacer le curseur à l'écran, au moyen de la souris, pour sélectionner, déplacer, copier ou supprimer des objets.
glissière	Partie immobile d'une commande ou d'un indicateur de face-avant qui contient des curseurs et des échelles.
glyphe	Petite image ou icône.
GPIB	*General Purpose Interface Bus*. C'est le nom courant de l'interface et du système de communication défini par les normes ANSI/IEEE 488.1-1987 et ANSI/IEEE 488.2-1987. Hewlett-Packard, inventeur de ce bus, l'appelle HP-IB.
graphe	*Voir* oscillographe, graphe déroulant et graphe à balayage.
graphe à balayage	Affichage similaire à celui d'un graphe oscillographe, à ceci près qu'une ligne verticale balaie l'affichage pour séparer les anciennes données des nouvelles.
graphe déroulant	Indicateur à tracé numérique modélisé d'après un enregistreur papier à défilement, qui s'enroule au fur et à mesure de l'enregistrement.
graphe oscillographe	Indicateur numérique inspiré du fonctionnement d'un oscilloscope.

H

handle	*Voir* pointeur.
hexa	Nombre hexadécimal (nombre exprimé en base 16).

housing	*Voir* glissière.

I

icon pane	*Voir* cadre icône.
icône	Représentation graphique d'un nœud sur un diagramme.
IEEE	*Institute for Electrical and Electronic Engineers*. Association d'ingénieurs qui établit des normes dans le domaine de l'électronique et de l'instrumentation.
impression contrôlée par programme	Impression automatique d'une face-avant d'un VI après son exécution.
indicateur	Objet de la face-avant qui sert à afficher les sorties.
Inf	Valeur d'affichage numérique qui désigne l'infini dans une représentation en virgule flottante.
inplace execution	*Voir* exécution "en place".
instrument virtuel	Programme LabVIEW. On l'appelle ainsi parce qu'il s'inspire de l'apparence et du fonctionnement d'un instrument réel.
I/O (Input/Output)	*Voir* E/S (Entrée/Sortie).
iteration terminal	*Voir* terminal d’itération

L

label	*Voir* étiquette.
Labeling tool	*Voir* outil Texte.
LabVIEW	*Laboratory Virtual Instrument Engineering Workbench.*
LED	*Light-emitting diode*. Diode électroluminescente (DEL).
légende	Objet appartenant à un graphe ou à un graphe déroulant, qui affiche son nom et précise les styles de tracés.

M

marquee	*Voir* marquise.

marquise	Contour mobile en pointillés qui entoure un objet sélectionné à l'écran.
matrice	Tableau à deux dimensions.
matrix	*Voir* matrice.
menus déroulants	Menus que l'on fait apparaître à l'écran à partir d'une barre de menus. Les options proposées dans ce type de menus portent habituellement sur des fonctionnalités générales.
menus locaux	Menus associés à des objets visualisés à l'écran. Pour faire apparaître un menu local, placez le curseur sur l'objet en question et cliquez sur le bouton droit de la souris (Windows, Sun et HP-UX) ou en maintenant la touche Commande enfoncée (Macintosh). Cette opération est appelée *pop-up* en anglais. *Voir pop-up*.
mise à l'échelle automatique	Possibilité qu'offrent les échelles de mesure de s'ajuster automatiquement à l'étendue des mesures affichées. Sur les graphiques, cette fonction détermine les valeurs maximale et minimale.
Mo	Méga-octets de mémoire.
mode Animation	Caractéristique qui anime l'exécution d'un VI pour illustrer le flux de données.

N

NaN	N'est-pas-un-nombre. Valeur d'affichage numérique pour la représentation à virgule flottante d'une variable qui *n'est pas un nombre*. C'est généralement le résultat d'une opération non définie telle que log(-1).
nœud	Elément de programme exécutable dans un diagramme. Ce peut être une fonction, une structure ou un sous-VI.
not-a-path	Valeur prédéfinie de la commande *Path* indiquant que le chemin n'est pas valide.
not-a-refnum	Valeur prédéfinie indiquant que le numéro de référence n'est pas valide.
numéro de référence de fichier	Numéro de référence qui sert à identifier un fichier ouvert. Le numéro de référence de fichier est utilisé par spécifier que vous voulez qu'une fonction ou qu'un VI effectue une opération sur le fichier ouvert.

O

objet	Terme générique utilisé pour désigner n'importe quel élément d'une face-avant ou d'un diagramme, tel qu'une commande, un nœud, un fil ou une image importée.
outil	Curseur particulier sous LabVIEW que l'on utilise pour effectuer des opérations à l'écran.
outil Bobine	Outil utilisé pour créer des chemins de données entre les terminaux source et de destination.
outil Doigt	Outil utilisé pour entrer des données dans une commande ou pour les faire fonctionner. Il est représenté par un index pointé.
outil Flèche	Outil utilisé pour déplacer et sélectionner des objets à l'écran et modifier leur taille.
outil Main	Outil utilisé pour faire défiler le contenu des fenêtres.
outil Menu local	Outil utilisé pour accéder au menu local d'un objet.
outil Pinceau	Outil utilisé pour colorier les objets et les arrière-plans.
outil Pipette	Outil utilisé pour recopier des couleurs des faces-avant.
outil Point d'arrêt	Outil utilisé pour définir un point d'arrêt sur un VI, un nœud ou un fil.
outil Sonde	Outil utilisé pour créer des sondes sur des fils.
outil Texte	Outil utilisé pour créer des étiquettes et entrer du texte dans les fenêtres de saisie.

P

palette	Menu qui affiche sous forme de palette des images représentant différentes options.
palette **Controls**	Palette contenant les commandes et les indicateurs de la face-avant.
palette **Functions**	Palette contenant les structures de diagrammes, les constantes, les fonctionnalités de communication et les VIs.
palette **Hierarchical**	Menu contenant des palettes et des sous-palettes.

palette **Tools**	Palette contenant des outils que vous pouvez utiliser pour modifier et mettre au point les objets des faces-avant et des diagrammes.
plate-forme	Ordinateur avec son système d'exploitation.
plot	*Voir* tracé.
point d'arrêt	Pause pendant l'exécution.
point de coercition	Glyphe sur un nœud ou un terminal, indiquant que la représentation numérique d'une donnée change en passant par ce point.
pointeur	Outil qui sert à accéder à un bloc de mémoire. Les pointeurs servent à travailler sur des tableaux ou des chaînes de caractères. Un tableau de chaînes de caractères sert à pointer un bloc de mémoire qui contient des pointeurs pour accéder aux chaînes de caractères.
polymorphisme	Possibilité offerte à un nœud de s'adapter automatiquement à des données de représentation, type et structure différents.
pop-up	Littéralement : faire surgir. Cela consiste à ouvrir un menu local en cliquant sur un objet avec le bouton droit de la souris (sous Windows, Sun et HP-UX) ou en maintenant la touche Commande enfoncée (sur Macintosh).
pop-up menus	*Voir* menus locaux.
Positioning tool	*Voir* outil Flèche.
probe	*Voir* sonde.
programmatic printing	*Voir* impression contrôlée par programme.
programmation séquentielle	Système de programmation par lequel l'ordre séquentiel d'écriture des instructions détermine l'ordre dans lequel elles sont exécutées. La plupart des langages de programmation textuels, tels que le C, le Pascal et le Basic, sont des langages de programmation séquentielle.
pull-down menus	*Voir* menus déroulants.

R

reentrant execution	*Voir* exécution réentrante.

registre à décalage	Mécanisme optionnel utilisé dans une structure de boucle pour faire passer la valeur d'une variable d'une itération de la boucle à l'itération suivante.
représentation	Sous-type de données numériques. Il existe des octets signés et non signés, des mots, des entiers longs, ainsi que des nombres à virgule flottante en simple précision, double précision ou précision étendue, qui peuvent être réels ou complexes.
resizing handles	*Voir* zone de redimensionnement.
ring control	*Voir* commande de type roue codeuse.
routine externe	*Voir* routine externe partagée.
routine externe partagée	Sous-programme qui peut être partagé par plusieurs ressources de code CIN.

S

scalaire	Nombre que l'on peut représenter par un point sur une échelle. Il s'agit d'une valeur singulière par opposition à un tableau regroupant plusieurs valeurs. Les variables scalaires, booléennes, chaînes de caractères et *clusters* sont des exemples particuliers des types de données respectifs.
scalar	*Voir* scalaire.
scale	*Voir* échelle.
scope chart	*Voir* oscilloscope.
sequence local	*Voir* variable locale de séquence.
Sequence Structure	*Voir* structure Séquence.
shared external routine	*Voir* routine externe partagée.
shell d'un *cluster*	Partie de la face-avant qui contient les objets d'un cluster.
shift register	*Voir* registre à décalage.
sink terminal	*Voir* terminal de destination.
slider	*Voir* curseur.

sonde	Fonction de mise au point qui sert à contrôler des valeurs intermédiaires lors de l'exécution d'un VI.
source terminal	*Voir* terminal source.
sous-diagramme	Diagramme à l'intérieur des bordures d'une structure.
sous-VI	VI utilisé dans le diagramme d'un autre VI. Il s'apparente à un sous-programme.
strip chart	*Voir* graphe déroulant.
structure	Elément de commande dans un programme, tel qu'une structure Séquence, une structure Condition, une boucle *For* ou une boucle *While*.
structure Condition	Structure de commande pour un branchement conditionnel qui aiguille le déroulement d'un programme vers un seul sous-diagramme en fonction de ses entrées. Cela correspond à une combinaison d'instructions IF, THEN, ELSE et CASE dans les langages de programmation séquentielle.
structure Séquence	Structure de programmation qui exécute ses sous-diagrammes suivant un ordre numérique. Elle sert souvent à imposer un ordre d'exécution à des nœuds dont le déclenchement n'est pas soumis à une dépendance de données.
sweep chart	*Voir* graphe à balayage.

T

table-driven execution	*Voir* exécution pilotée par tableau.
tableau	Ensemble de données d'un même type, triées et indexées.
tableau vide	Tableau pour lequel on a défini un type de données, mais qui ne contient pas d'éléments. Il s'agit par exemple d'un tableau qui compte une commande numérique dans sa fenêtre d'affichage des données, mais qui n'a pas reçu de valeur pour aucun de ses éléments.
tableau vierge	Objet de la face-avant qui contient un tableau. Il comprend un tableau indexé, une fenêtre présentant des données et une étiquette optionnelle. Il peut recevoir des données de divers types.
terminal	Objet ou zone d'un nœud que traversent les données.

terminal conditionnel	Terminal d'une boucle *While* qui contient une valeur booléenne déterminant si le VI doit ou non exécuter une itération supplémentaire de la boucle.
terminal de comptage	Variable liée à une boucle *For* dont la valeur détermine le nombre de fois que la boucle doit exécuter son sous-diagramme.
terminal de destination	*Voir* terminal d'entrée.
terminal d'entrée	Terminal qui absorbe les données. Synonyme de terminal de destination.
terminal d'itération	Terminal d'une boucle *For* ou d'une boucle *While*, qui contient le nombre d'itérations déjà effectuées à un instant donné.
terminal source	Terminal qui émet des données.
texte libre	Etiquette sur la face-avant ou le diagramme qui n'appartient à aucun autre objet.
tool	*Voir* outil.
top-level VI	*Voir* VI principal.
tracé	Représentation graphique d'un tableau de données, sous forme de graphe ou de graphe déroulant.
tunnel	Terminal d'entrée ou de sortie dans une structure.

V

V	Volts.
variable globale	Sous-VI non réentrant (c'est-à-dire qui ne peut pas s'appeler lui-même) doté de mémoire locale qui utilise un registre à décalage non initialisé, ce qui permet de conserver des données entre deux exécutions. La mémoire des copies de ces sous-VIs est partagée et peut ainsi servir au transfert des variables globales entre eux.
variable locale de séquence	Terminal qui transmet les données entre les cadres d'une structure Séquence.
VI	*Virtual Instrument*. *Voir* instrument virtuel.
VI à mauvaise connexion	Prototype non fonctionnel d'un sous-VI créé par l'utilisateur. Bien qu'incluant des entrées et des sorties, il est incomplet. Il sert de marque aux premiers stades de la conception des VIs pour le développement de futurs VIs.

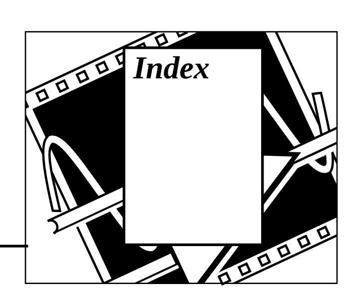
VI en cours	VI dont la face-avant, le diagramme ou la fenêtre d'Editeur d'icônes constitue la fenêtre active.
VI invalide	VI qui ne peut pas être compilé ou exécuté. Il est symbolisé par une flèche brisée sur le bouton Exécution.
VI principal	VI qui se trouve au sommet de la hiérarchie des VIs. Ce terme permet de distinguer les VIs de leurs sous-VIs.

W

wire	*Voir* fil.
Wiring tool	*Voir* outil Bobine.

Z

zone de redimensionnement	Petit repère angulaire situé dans les coins d'un objet, qui sert à indiquer les points où il faut cliquer pour modifier la taille de l'objet.



Index

A

B

D

E

F

G

H

M

N

O

S

T

U

V

W